An Implicit, Nonlinear Reduced Resistive MHD Solver

L. Chacón,[a] **D. A. Knoll,**[b] **J. M. Finn**[c]

August 8, 2000

[a] MS K717, PO Box 1663, Los Alamos National Laboratory, Los Alamos, NM 87545, e-mail: chacon@lanl.gov

[b] MS B216, PO Box 1663, Los Alamos National Laboratory, Los Alamos, NM 87545, e-mail: nol@lanl.gov

[c] MS K717, PO Box 1663, Los Alamos National Laboratory, Los Alamos, NM 87545, e-mail: finn@lanl.gov

Running head: Implicit Resistive MHD Solver

Corresponding author: Luis Chacón, P.O. Box 1663, MS K717, Los Alamos National Laboratory, Los Alamos, NM 87545; voice (505) 665-6973, fax (505) 665-7150); e-mail: chacon@lanl.gov

Abstract

Implicit time differencing of the resistive magnetohydrodynamic (MHD) equations can step over the limiting time scales –such as Alfvén time scales– to resolve the dynamic time scales of interest. However, nonlinearities present in these equations make an implicit implementation cumbersome. Here, viable paths for an implicit, nonlinear time integration of the MHD equations are explored using a 2D reduced viscous-resistive MHD model. The implicit time integration is performed using the Newton-Raphson iterative algorithm, employing Krylov iterative techniques for the required algebraic matrix inversions, implemented Jacobian-free (i.e., without ever forming and storing the Jacobian matrix). Convergence in Krylov techniques is accelerated by preconditioning the initial problem. A "physics-based" preconditioner, based on an operator-split approximation to the original set of partial differential equations, is employed. The preconditioner employs low-complexity multigrid techniques to invert approximately the resulting elliptic algebraic systems. The resulting 2D reduced resistive MHD implicit algorithm is shown to be successful in dealing with large time steps (100-250 times the explicit Alfvén CFL limit) and fine grids (up to 256x256). The algorithm is second-order accurate in time and efficient. Specifically, the number of Krylov iterations per Newton iteration scales very weakly with the total number of mesh points, and the CPU time scales as $CPU \sim \Delta t^{-0.7}$.

Keywords: Jacobian-free, Newton-Krylov, multigrid, physics-based preconditioning, implicit differencing, resistive MHD.

AMS subject classification: 65C20, 65M06, 65P20, 77A05

1 Introduction

The resistive magnetohydrodynamic (MHD) equations present a formidable challenge for efficient implicit differencing due to the disparity of time scales, the nonlinear couplings present in the equations, and fine grids needed to resolve current sheets. Partially implicit (or semi-implicit) time-differencing schemes have been employed ubiquitously in multidimensional calculations, in which only part of the equations is integrated implicitly. Among such techniques we find alternating direction implicit (ADI) methods [1, 2] and specific semi-implicit techniques for MHD [3, 4, 5, 6]. In the former, the integration is implicit in one direction, and explicit in the others, alternating directions every time step. ADI methods do not render the numerical integration absolutely stable for arbitrary time steps, but allow time steps $\sim 10 - 20$ times larger than the explicit Courant-Friedrichs-Lewy (CFL) stability limit Δt_{CFL} [1, 2].

Within the MHD community, semi-implicit methods have been developed in which the discretized set of partial differential equations (PDE's) is modified by adding implicitly and subtracting explicitly some simple, easily invertible operator (called semi-implicit operator). This renders the numerical method unconditionally stable for arbitrary time steps while requiring much less work than a direct implicit integration. The semi-implicit operator accomplishes this by slowing down the propagation of the fastest waves in the problem. However, stability does not guarantee accuracy. The accuracy of the numerical solution obtained with MHD semi-implicit techniques is strongly dependent on the choice of the form of the semi-implicit operator [5, 6], and on the size of the time step employed, often requiring time-consuming convergence studies. As a result, the time step in semi-implicit methods is typically limited by *accuracy* considerations to $\gamma\Delta t \leq 0.05$ [5], where γ^{-1} determines the time scale of interest.

There has been a recent attempt to implement a nonlinear, implicit difference scheme for the MHD equations [7]. This approach employs block Gauss-Seidel techniques to invert an approximate Jacobian matrix (and LU-decomposition techniques to invert the blocks) to obtain convergence including the nonlinear terms in the equations. However, the simplifications in the Jacobian (required to render it manageable) limit the applicability of the method, because they introduce CFL restrictions for large and moderate viscosities and resistivities. Furthermore, the nonlinear residual is never used to check nonlinear convergence (the size of the nonlinear update δx is used for convergence), and hence nonlinear errors stemming from the approximate Jacobian remain unmonitored.

None of these methods employ modern approaches to deal with the grid refinement stiffness (such as multigrid methods), and none monitor the nonlinear residual to ensure nonlinear convergence at the implicit time level. The present document explores viable paths for fully implicit, fully nonlinear differencing of the MHD equations. For this purpose, a 2D reduced resistive MHD model, supporting shear Alfvén and sound waves, is employed. Two-fluid effects (key in collisionless reconnection processes [8, 9]) will be considered in a second stage of the research, once the fundamental approach to the –simpler– resistive problem is mature. Convergence in the nonlinear system is achieved using the Newton-Raphson iterative algorithm. Krylov iterative techniques [10], implemented Jacobian-free [11, 12] (i.e., without ever forming and storing the Jacobian matrix), are employed for the required algebraic matrix inversions, because of their efficiency and the possibility of accelerating convergence via preconditioning. Here, PGMRES (*p*reconditioned *g*eneralized *m*inimal *res*iduals [13]) is employed due to the lack of symmetry in the algebraic system.

The efficiency of Krylov methods depends heavily on adequate preconditioning [10, 14]. Here, a "physics-based" (or PDE-based) preconditioner [12] –in which simplifications of the original system of PDE's are dictated by knowledge of the physical system– is explored. Diagonal blocks in the preconditioner are inverted using low-complexity multigrid methods (MG) [11, 15, 16] to remove the grid stiffness. In order to deal with the lack of diagonal dominance characteristic of algebraic systems associated with wave propagation operators, the method of differential approximation [17] is employed to modify the original set of difference equations to render the algebraic system diagonally dominant, while preserving its wave propagation properties.

The rest of the paper is organized as follows. Section 2 introduces the base model equations. Section 3 introduces the Krylov methods and the specifics of the Jacobian-free implementation. In Sec. 4, the "physics-based" preconditioner for this particular application is developed. Validation and efficiency results of the implicit algorithm are presented in Sec. 5, where it is numerically demonstrated that:

1. It performs nearly optimally under grid refinement (the number of Krylov iterations scales very weakly with N, the total number of mesh points).

2. It presents a sublinear scaling of the number of Krylov iterations with the implicit time step ($\sim \Delta t^{0.3}$), which results in a CPU time scaling $\sim \Delta t^{-0.7}$.

3. It features second-order accurate time step convergence.

Finally, we conclude in Sec. 6.

2 2D Reduced MHD model

In the 2D reduced MHD (RMHD) formalism, the magnetic field component in the ignorable direction B_z is much larger than the magnitude of the poloidal magnetic field $\vec{B}_p$. As a result, $B_z \approx$ constant and the poloidal velocity $\vec{v}_\perp$ is incompressible ($\nabla_\perp \cdot \vec{v}_\perp = 0$, with $\nabla_\perp$ the poloidal gradient), and the general MHD formalism reduces to [18, 19]:

$$\begin{aligned}
\nabla_\perp^2 \Phi &= \omega & (1)\\
(\partial_t + \vec{v}_\perp \cdot \nabla_\perp - \frac{\eta}{\mu_0}\nabla_\perp^2)\Psi + E_0 &= 0 & (2)\\
\rho(\partial_t + \vec{v}_\perp \cdot \nabla_\perp - \nu\nabla_\perp^2)v_{||} + \dot{S}_{v_{||}} &= -\vec{b}\cdot\nabla p & (3)\\
(\partial_t + \vec{v}_\perp \cdot \nabla_\perp - D\nabla_\perp^2)\rho + \dot{S}_\rho &= -\vec{b}\cdot\nabla(\rho v_{||}) & (4)\\
\rho(\partial_t + \vec{v}_\perp \cdot \nabla_\perp - \nu\nabla_\perp^2)\omega + \dot{S}_\omega &= \frac{1}{\mu_0}\vec{B}\cdot\nabla(\nabla_\perp^2\Psi) + 2(\vec{z}\times\vec{\kappa}_c)\cdot\nabla p & (5)
\end{aligned}$$

where Φ is the poloidal velocity stream function ($\vec{v}_\perp = \vec{z} \times \nabla\Phi$), ω is the vorticity in the poloidal plane ($\omega = \vec{z} \cdot \nabla \times \vec{v}_\perp$), Ψ is the poloidal flux function (which gives $\vec{B}_p = \vec{z} \times \nabla\Psi$), and $v_{||}$ is the parallel velocity ($\vec{v} = \vec{v}_\perp + v_{||}\vec{b}$, with $\vec{b} = \vec{B}/B \approx \vec{B}/B_z$). Sources E_0 (the applied electric field in the z direction), $\dot{S}_\omega$, $\dot{S}_\rho$, and $\dot{S}_{v_{||}}$ have been included to balance the decay of the equilibrium due to transport terms. The transport parameters (the kinematic viscosity ν, the resistivity η, and the cross-field particle diffusion coefficient D) are assumed constant; the viscosity coefficient in Eq. 3 has been chosen for convenience as isotropic and equal to the perpendicular viscosity (Eq. 5).

The pressure is evaluated as $p = \rho T/m_i$, with m_i the ion mass and T the temperature. The temperature is assumed constant. It is assumed further that the Boussinesq approximation applies [i.e., $\rho(\vec{r}) = \rho_0 + \tilde{\rho}(\vec{r})$ with $\rho_0 \gg \tilde{\rho}(\vec{r})$], so that $\rho \approx \rho_0$ in the inertial terms of Eqs. 3 and 5. In Eq. 5, μ_0 is the vacuum permeability constant, and a curvature term has been included $(2(\vec{z} \times \vec{\kappa}_c) \cdot \nabla p)$ [18, 19], with $\vec{\kappa}_c = \vec{b} \cdot \nabla\vec{b}$ the field line curvature. According to the previous approximations, and for a 2D model ($\partial_z = 0$), the field line curvature reads $\vec{\kappa}_c \approx \frac{(\vec{B}_p \cdot \nabla_p)\vec{B}_p}{B_z^2}$. In typical configurations of interest (cylinder, torus), the plasma poloidal cross section is circular, and the poloidal magnetic field is predominantly oriented in the angular direction $\hat{\theta}$. Hence, $\vec{B}_p \approx B_p\hat{\theta}$, and accordingly $\vec{\kappa}_c \approx -\frac{B_p^2}{rB_z^2}\hat{r}$, where $\hat{r}$ is the radial unit vector.

Resistive reconnection effects are important only in the vicinity of a rational surface (at which $\vec{B}_p = 0$), and thus, in cylindrical geometry, only an annulus surrounding the rational surface is considered. This annulus is assumed to be much thicker than the resistive layer, but sufficiently thin (i.e., $\Delta r_{an} \ll r_{rs}$, see Fig. 1) so that a rectangular approximation of the circular layer is satisfactory (the effect of the cylindrical geometry is the inclusion of the curvature term). In this approximation, the cylindrical coordinates are approximated by Cartesian coordinates by identifying $dr \to dy$ and $rd\theta \to -dx$, the computational rectangle is defined by $L_x = 2\pi r_{rs}$, $L_y = \Delta r_{an}$ ($L_x \gg L_y$), and the poloidal field curvature reads:

$$\vec{\kappa}_c \approx -\frac{2\pi B_p^2}{L_x B_z^2}\hat{y} \tag{6}$$

Finally, in Eqs. 1-5, B_p is normalized to the poloidal magnetic field at the wall in equilibrium, ρ to ρ_0, $v_\perp$ to the poloidal Alfvén speed $v_A = \sqrt{B_p^2/\rho_0\mu_0}$, $v_{||}$ to the ion sound speed $c_s = \sqrt{\frac{T}{m_i}}$, lengths to the characteristic length in the y-direction L_y, and the time to the poloidal Alfvén time $\tau_A = L_y/v_A$. The normalized set of RMHD equations reads:

$$\begin{aligned}
\nabla_\perp^2\Phi &= \omega & (7)\\
(\partial_t + \vec{v}_\perp\cdot\nabla_\perp - \frac{1}{S_L}\nabla_\perp^2)\Psi + E_0 &= 0 & (8)\\
(\partial_t + \vec{v}_\perp\cdot\nabla_\perp - \frac{1}{Re}\nabla_\perp^2)v_{||} + \dot{S}_{v_{||}} &= -\sqrt{\frac{\beta}{2}}\vec{B}\cdot\nabla\rho & (9)\\
(\partial_t + \vec{v}_\perp\cdot\nabla_\perp - D\nabla_\perp^2)\rho + \dot{S}_\rho &= -\sqrt{\frac{\beta}{2}}\vec{B}\cdot\nabla(\rho v_{||}) & (10)\\
(\partial_t + \vec{v}_\perp\cdot\nabla_\perp - \frac{1}{Re}\nabla_\perp^2)\omega + \dot{S}_\omega &= \vec{B}\cdot\nabla(\nabla_\perp^2\Psi) + \frac{2\pi\beta}{L_x}\frac{\partial\rho}{\partial x} & (11)
\end{aligned}$$

where $S_L = \frac{\mu_0 L_y v_A}{\eta}$ is the Lundquist number (or magnetic Reynolds number), $Re = \frac{L_y v_A}{\nu}$ is the Reynolds number, and $\beta = \frac{2\mu_0 p}{B_z^2}$ is the total plasma beta. All magnitudes in the equations are dimensionless at this point.

Equations 7-11 are discretized in time employing a second-order accurate Crank-Nicolson difference scheme, as follows:

$$\begin{aligned}
\nabla_\perp^2\Phi^{n+1} &= \omega^{n+1} & (12)\\
\frac{\Psi^{n+1}-\Psi^n}{\Delta t} + \nabla_\perp\cdot(\vec{v}_\perp\Psi)^{n+\frac{1}{2}} - \frac{\nabla_\perp^2\Psi^{n+\frac{1}{2}}}{S_L} &= -E_0 & (13)\\
\frac{v_{||}^{n+1}-v_{||}^n}{\Delta t} + \nabla_\perp\cdot(\vec{v}_\perp v_{||})^{n+\frac{1}{2}} - \frac{\nabla_\perp^2 v_{||}^{n+\frac{1}{2}}}{Re} &= -\sqrt{\frac{\beta}{2}}(\vec{B}\cdot\nabla\rho)^{n+\frac{1}{2}} - \dot{S}_{v_{||}} & (14)\\
\frac{\rho^{n+1}-\rho^n}{\Delta t} + \nabla_\perp\cdot(\vec{v}_\perp\rho)^{n+\frac{1}{2}} - D\nabla_\perp^2\rho^{n+\frac{1}{2}} &= -\sqrt{\frac{\beta}{2}}[\vec{B}\cdot\nabla(\rho v_{||})]^{n+\frac{1}{2}} - \dot{S}_\rho & (15)\\
\frac{\omega^{n+1}-\omega^n}{\Delta t} + \nabla_\perp\cdot(\vec{v}_\perp\omega)^{n+\frac{1}{2}} - \frac{\nabla_\perp^2\omega^{n+\frac{1}{2}}}{Re} &= [\vec{B}\cdot\nabla(\nabla_\perp^2\Psi)]^{n+\frac{1}{2}} + \frac{2\pi\beta}{L_x}\frac{\partial\rho^{n+\frac{1}{2}}}{\partial x} - \dot{S}_\omega & (16)
\end{aligned}$$

where quantities at the $n+\frac{1}{2}$ time level are calculated as $\xi^{n+\frac{1}{2}} = \theta\xi^n + (1-\theta)\xi^{n+1}$, with θ a shifting parameter [$0 \leq \theta < 0.5$ for the method to remain implicit; $\theta = 0$ is backward Euler, $\theta = 0.5$ achieves a second order accuracy in time, and $\theta = 1$ is forward Euler (explicit)]. In the discretization above, it has been taken into account that $\nabla_\perp \cdot \vec{v}_\perp = 0$ to put advection terms in conservative form. Spatial operators are discretized using second-order centered finite differences.

Notice that the first equation in the previous set of difference equations does not involve a time step, and represents a purely elliptic constraint. This constraint has to be satisfied to the prescribed tolerance independently of the time step chosen, thus imposing additional strain on the solver.

Although the time-stepping algorithm is absolutely stable regardless of time step size due to the implicit differencing, accuracy considerations still limit the maximum time step size. Crank-Nicolson with $\theta = 0.5$ is second order accurate, but it is known to "ring" (i.e., to propagate weakly damped short-wavelength harmonics) whenever the implicit time step is much larger than the Courant (diffusion) explicit time step limit. Large implicit time steps are still possible with $\theta \sim 0.48-0.49$, which provides sufficient damping while approximately preserving the second-order accuracy. We will demonstrate numerical second-order accuracy in time for $\theta = 0.5$ in Sec. 5.3.1.

3 Jacobian-free Newton-Krylov solver

Once the RMHD equations are discretized in time and space, the problem boils down to finding the new-time solution $\vec{x}^{n+1} = \{\vec{\Phi}^{n+1}, \vec{\Psi}^{n+1}, \vec{v}_{||}^{n+1}, \vec{\rho}^{n+1}, \vec{\omega}^{n+1}\}^T$ from the current-time solution $\vec{x}^n$ by solving the nonlinear, coupled system of equations in Eqs. 12-16, symbolized by $\vec{G}(\vec{x}^{n+1}) = \vec{0}$ (where $\vec{G} = \{\vec{G}_\Phi, \vec{G}_\Psi, \vec{G}_{v_{||}}, \vec{G}_\rho, \vec{G}_\omega\}^T$). This is accomplished iteratively with the Newton-Raphson algorithm, which requires the solution of a series of algebraic systems of the form:

$$J_k \delta\vec{x}_k = -\vec{G}(\vec{x}_k) \tag{17}$$

Here, $J_k = \left(\frac{\partial\vec{G}}{\partial\vec{x}}\right)_k$ is the Jacobian matrix, $\vec{x}_k$ is the kth state vector, $\delta\vec{x}_k$ is the kth Newton update (from which the (k+1)th Newton state vector is obtained, $\vec{x}_{k+1} = \vec{x}_k + \delta\vec{x}_k$), $\vec{G}(\vec{x}_k)$ is the vector of residuals, and k is the nonlinear iteration level. Nonlinear convergence is achieved when:

$$\left\|\vec{G}(\vec{x}_k)\right\|_2 < \epsilon_{newton} \left\|\vec{G}(\vec{x}_0)\right\|_2 \tag{18}$$

where $\|\cdot\|_2$ is the L_2-norm, ϵ_{newton} is the Newton convergence tolerance (set to 10^{-4} in this work), and $\vec{G}(\vec{x}_0)$ is the initial residual. Upon convergence, the solution at the new time step is found as $\vec{x}^{n+1} = \vec{x}_{k+1}$.

Each of these iterative steps requires inverting the Jacobian system in Eq. 17. This is performed here using Krylov methods [10], because:

1. As stand-alone solvers, they are already competitive against direct or standard iterative techniques, and have the capability of increased efficiency via adequate preconditioning.

2. All these methods require to proceed is the product of the system matrix times a Krylov vector $\vec{v}$, which is provided by the iterative algorithm. In the case of the Jacobian system above, the Jacobian-vector product can be approximated by the following first-order difference formula:

$$J_k\vec{v} \approx \frac{\vec{G}(\vec{x}_k + \epsilon\vec{v}) - \vec{G}(\vec{x}_k)}{\epsilon} \tag{19}$$

   where ϵ is small but finite. Thus, the evaluation of the Jacobian-vector product only requires the function evaluation $\vec{G}(\vec{x}_k + \epsilon\vec{v})$. There is no need to form or store the Jacobian matrix, and hence the name Jacobian-free.

3. An additional advantage of not forming the Jacobian matrix is that cumbersome d-ifference schemes (such as conservative and/or high-order difference schemes) are of straightforward implementation and maintenance.

Among the various Krylov methods available, PGMRES is selected because it guarantees convergence with nonsymmetric, nonpositive definite systems [13] (the case here because of flow and wave propagation), and because it provides normalized Krylov vectors $|\vec{v}| = 1$, thus bounding the error introduced in the difference approximation of Eq. 19 (whose leading error term is proportional to $\epsilon \mid \vec{v} \mid^2$) [20]. However, PGMRES can be memory intensive (storage increases linearly with the number of PGMRES iterations per Jacobian solve) and expensive (computational complexity of GMRES increases with the square of the number of PGMRES iterations per Jacobian solve). Thus, minimizing the number of PGMRES iterations per Jacobian solve is crucial for efficiency. This is accomplished here in two ways: 1) using inexact Newton techniques [21], and 2) improving the condition number of the Jacobian matrix by preconditioning the problem.

The inexact Newton method adjusts the PGMRES convergence tolerance at every Newton iteration according to the size of the Newton residual, as follows:

$$\left\| J_k \delta \vec{x}_k + \vec{G}(\vec{x}_k) \right\|_2 < \zeta \left\| \vec{G}(\vec{x}_k) \right\|_2 \tag{20}$$

where ζ is the inexact Newton parameter. Thus, the convergence tolerance of PGMRES is loose when the state vector $\vec{x}_k$ is far from the nonlinear solution, but becomes increasingly tighter as $\vec{x}_k$ approaches the exact solution. Hence, PGMRES works hard only when the state vector is close to the exact solution. While the characteristic quadratic convergence rate of Newton's method is typically lost with inexact Newton techniques, the gain in PGMRES performance typically overcomes the slight decrease of the Newton convergence rate (which nevertheless remains superlinear). The sensitivity of the number of PGMRES iterations to ζ is shown in Sec. 5.2.3. Values of $\zeta \sim 5 \cdot 10^{-2}$ work well for this application.

Preconditioning consists in operating on the system matrix J_k with an operator P_k (preconditioner) such that the condition number of $J_k P_k$ is close to unity (i.e., $J_k P_k \approx I$, the identity matrix). The Jacobian-free implementation of the right preconditioner operator is straightforward when considering the equivalent system:

$$(J_k P_k)(P_k^{-1}\delta\vec{x}_k) = -\vec{G}(\vec{x}_k) \tag{21}$$

Thus, PGMRES will solve:

$$(J_k P_k)\vec{z} = -\vec{G}(\vec{x}_k) \tag{22}$$

and the Newton update $\delta\vec{x}_k$ is found upon obtaining $\vec{z}$ by finding $\delta\vec{x}_k = P_k\vec{z}$. Notice that the system in Eq. 21 is equivalent to the original system for any nonsingular operator P_k. Thus, the choice of P_k does not affect the accuracy of the final solution; however, it crucially determines the efficiency of the algorithm.

To solve Eq. 22 using PGMRES, it is required to compute the Jacobian-vector product $(J_k P_k)\vec{v}_j$ (where $\vec{v}_j$ is the Krylov vector of the jth iteration) to proceed. This is implemented in two steps:

1. Compute $\vec{y} = P_k\vec{v}_j$. This is the so-called preconditioner step. Often, P_k is not an exact inverse of any particular matrix, but an approximate inverse –obtained, for instance, using operator splitting and/or low-complexity MG methods– of the exact Jacobian, or even an approximate inverse of an approximation of the Jacobian. The specifics of the formation of the preconditioner operator P_k for this application are discussed in Sec. 4.

2. Compute $J_k\vec{y}$ using the Jacobian-free approximation: $J_k\vec{y} \approx \frac{\vec{G}(\vec{x}_k+\epsilon\vec{y})-\vec{G}(\vec{x}_k)}{\epsilon}$, where ϵ is small but finite. Following Refs. [22] and [23], ϵ is calculated here as:

$$\epsilon = 10^{-5}\frac{\|\vec{x}_k\|_2}{N\,\|\vec{y}\|_2}$$

where $N = N_x \text{x} N_y$ is the total number of mesh points.

The first step determines the efficiency of the algorithm (and leaves room for exploration, since P_k is in principle an arbitrary nonsingular operator), while the second step determines the accuracy of the solution (according to the discretization of the nonlinear system $\vec{G}(\vec{x}^{n+1}) = \vec{0}$). The rest of the paper will discuss the techniques employed in the first step.

To maximize efficiency, the preconditioning operator P_k should approximate the inverse of the Jacobian J_k while being relatively inexpensive. There are generally two choices as to how to approach the preconditioning problem:

1. *Algebraic methods*: they approximately invert a close representation of the Jacobian J_k, obtained analytically or numerically, using inexpensive algebraic techniques (such as stationary iterative techniques, incomplete Cholesky decomposition, multigrid techniques, etc.). These techniques are "problem-independent" (can be employed in a variety of different problems), but by the same token they cannot exploit any specific knowledge of the problem at hand. In addition, they typically require forming and storing the Jacobian matrix (although forming it once every 5 to 10 Newton iterations has proven to work well [24]).

2. *PDE-based or physics-based methods*: they approximately integrate a simplified set of PDE's (obtained for instance, by a simple Picard linearization and/or by operator splitting [12]). They do not require forming and storing the complete Jacobian, and hence take better advantage of the Jacobian-free implementation than algebraic methods. In addition, they can be optimized for the problem at hand. Although the general concept of physics-based preconditioning can be applied to a large variety of problems, the details of the implementation are typically "problem-specific."

The latter is the choice here, because it leads to a "distributed" solution, in which only diagonal blocks need be inverted (see next section). This typically results in more efficient preconditioners [25], because: 1) while diagonal blocks can be easily constructed to be diagonally dominant because they contain elliptic operators, the complete system is usually not diagonally dominant because it contains wave propagation operators in off-diagonal blocks; and 2) inverting m blocks of dimension NxN (with m the number of equations) is less expensive than inverting a single mNxmN matrix. In addition, the implementation of MG techniques for scalar problems is less cumbersome than for systems of equations. The next section describes in more detail the nature of the approximations employed here to form a physics-based preconditioner.

4 "Physics-based" preconditioner

Implicit differencing ensures absolutely stable numerical descriptions, for any time step and level of mesh refinement, by introducing dispersion in waves and by treating elliptic operators (such as diffusion) nonlocally. However, some of the mechanisms that are source of numerical instabilities in explicit methods continue to manifest themselves in implicit schemes in the form of ill-conditioned algebraic systems, which iterative techniques have difficulty in handling.

There are two sources of stiffness in the system of MHD equations: grid stiffness and wave stiffness. The grid stiffness stems from poorly conditioned matrices resulting from the discretization of elliptic operators, and manifests itself in a power scaling N^{α} (where $\alpha > 0$) of the computational complexity with the number of mesh points N [26]. It is dealt with here by multigrid preconditioning (MG), which employs low-complexity multilevel solvers [16, 11] to invert the elliptic operators approximately. The multilevel aspect of MG (which employs a "divide and conquer" approach by which the different scales of the global solution are decoupled in multiple grids of varying mesh refinement) typically results in a number of Krylov iterations virtually independent of the problem size (see results in Sec. 5.3.2).

Wave stiffness also results in ill-conditioned algebraic systems, typically manifesting itself in a loss of diagonal dominance due to short-wavelength harmonics when the implicit time step is larger than the explicit wave CFL limit (as a side note, short-wavelength harmonics are also responsible for instability in explicit methods). As a consequence, wave-propagating algebraic systems are handled poorly by iterative techniques, which typically perform better with diagonal dominance.

The system of PDE's in Eqs. 7-11 support propagation of two waves: the shear Alfvén wave (embodied in Eqs. 7, 8, and 11), and the sound wave (Eqs. 9 and 10). After linearization, and for the equation ordering chosen in Eqs. 12-16, the Jacobian matrix J_k of this system of equations has the following block-matrix structure:

$$J_k = \begin{bmatrix} D_\Phi & 0 & 0 & 0 & I \\ L_{\Phi,\Psi} & D_\Psi & 0 & 0 & 0 \\ L_{\Phi,v_{||}} & L_{\Psi,v_{||}} & D_{v_{||}} & U_{\rho,v_{||}} & 0 \\ L_{\Phi,\rho} & L_{\Psi,\rho} & L_{v_{||},\rho} & D_\rho & 0 \\ L_{\Phi,\omega} & L_{\Psi,\omega} & 0 & L_{\rho,\omega} & D_\omega \end{bmatrix} \tag{23}$$

Each row of blocks corresponds to a discretized equation. In Eq. 23, I is the identity block, the D-blocks are the block diagonals [D_Φ contains the discretization of $\nabla_\perp^2$, the blocks $D_{\Psi,v_{||},\omega}$ contain the discretization of $(\partial_t + \vec{v}_\perp \cdot \nabla_\perp - \lambda\nabla_\perp^2)$, with $\lambda = \{S_L^{-1}, Re^{-1}\}$ depending on the equation, and D_ρ contains the discretization of $(\partial_t + \vec{v}_\perp \cdot \nabla_\perp - D\nabla_\perp^2 + \sqrt{\frac{\beta}{2}}\vec{B}_0 \cdot \nabla(v_{||,0}) + \sqrt{\frac{\beta}{2}}v_{||,0}\vec{B}_0 \cdot \nabla)$], and the L-blocks and U-blocks represent couplings between dependent variables. For the equation ordering chosen, J_k is almost a lower-triangular block matrix, except for two off-diagonal upper blocks: $U_{\rho,v_{||}}$ (which corresponds to the $-\sqrt{\frac{\beta}{2}}\vec{B}_0 \cdot \nabla\delta\rho$ term in the linearized form of Eq. 9, and is required for the sound wave), and the identity block I (which corresponds to $\delta\omega$ in the linearization of Eq. 7, and is required for the Alfvén wave). For large implicit time steps and small transport coefficients λ, the derivatives of the shortest wavelength harmonics supported by the grid (i.e., with the largest wavenumber) will result in some off-diagonal blocks dominating over the diagonal blocks, and simple block iterative techniques (such as block symmetric Gauss-Seidel) will fail. Time step sub-cycling in the preconditioner (so that the time step is close to the explicit CFL limit) would work, but at the cost of efficiency.

Clearly, an alternative route must be found to form a valid preconditioner. One such approach is to reformulate the physical equations so that the resulting algebraic systems are diagonally dominant for any wavelength harmonics, despite the wave propagation terms. A rigorous technique for such transformations is the so-called method of differential approximation. This is explained further in the next section.

4.1 Method of differential approximation

The method of differential approximation [17] is a rigorous, systematic way of deriving a new set of PDE's from the time-difference formulation of the original set of PDE's. The new set of PDE's satisfies several properties:

1. After differencing, the resulting discrete equations are consistent with the original set of PDE's to the same order of accuracy as the original set of difference equations.

2. They have the same steady-state solution as the original set of difference equations.

3. The additional terms in the new set of PDE's are located in the diagonal blocks, and result in block-diagonally dominant systems.

As mentioned earlier, the system of PDE's in Eqs. 7-11 support propagation of two waves: the shear Alfvén wave and the sound wave, whose associated time scales are usually orders of magnitude smaller than the dynamic time scales of interest. These are discussed next.

4.1.1 Sound wave

The sound wave is propagated by the density equation (Eq. 10) and the parallel velocity equation (Eq. 9). After linearization ($v_{||} = v_{||,0} + \delta v_{||}$ and $\rho = \rho_0 + \delta\rho$, where the subscript "0" indicates the equilibrium, and the "δ" indicates the perturbations) for uniform density and magnetic field and no poloidal flow, the sound wave propagation terms are:

$$\partial_t \delta v_{||} = -\sqrt{\frac{\beta}{2}} \vec{B}_0 \cdot \nabla \delta\rho \tag{24}$$

$$\partial_t \delta\rho = -\sqrt{\frac{\beta}{2}} \vec{B}_0 \cdot \nabla \delta v_{||} \tag{25}$$

Other terms in the linearization are neglected because the goal is not accuracy, but producing a diagonally dominant formulation of the wave equations. A second-order time-implicit differencing of these equations yields:

$$\begin{aligned} \frac{\delta v_{||}^{n+1} - \delta v_{||}^{n}}{\Delta t} &= -\sqrt{\frac{\beta}{2}} \vec{B}_0 \cdot \nabla \delta\rho^{n+\frac{1}{2}} \\ \frac{\delta \rho^{n+1} - \delta \rho^{n}}{\Delta t} &= -\sqrt{\frac{\beta}{2}} \vec{B}_0 \cdot \nabla \delta v_{||}^{n+\frac{1}{2}} \end{aligned}$$

Taylor-expanding the time difference of $\delta v_{||}$, we find:

$$\frac{\delta v_{||}^{n+1} - \delta v_{||}^{n}}{\Delta t} \approx \left[\partial_t \delta v_{||} + \frac{\Delta t^2}{24} \partial_{ttt} \delta v_{||}\right]^{n+\frac{1}{2}}$$

Using the sound wave propagation equations above, we can write:

$$\partial_{tt} \delta v_{||} = \frac{\beta}{2} (\vec{B}_0 \cdot \nabla)^2 \delta v_{||}$$

and hence the difference form of $\partial_t \delta v_{||}$ can be rewritten as:

$$\left[\partial_t \delta v_{||}\right]^{n+\frac{1}{2}} \approx \left[1 - \kappa \Delta t^2 \frac{\beta}{2} (\vec{B}_0 \cdot \nabla)^2\right] \frac{\delta v_{||}^{n+1} - \delta v_{||}^{n}}{\Delta t} \tag{26}$$

where the constant $\frac{1}{24}$ coming from the truncation error analysis has been replaced by a parameter κ (a sensitivity analysis of the number of PGMRES iterations in terms of κ is presented in Sec. 5.2.2). The difference form in Eq. 26 is to be used for $\partial_t \delta v_{||}$ in Eq. 24 instead of the standard centered finite difference formula. The same procedure can be followed for the difference form of $\partial_t \delta\rho$ in Eq. 25. However, including the wave-correction term only in one of the wave equations has proven to work well in practice [3, 5, 17]. Here, the $\delta v_{||}$-equation has been chosen for reasons that will become apparent shortly.

Several comments on Eq. 26 are in order:

1. The additional term (subsequently called "wave-correction operator") in the difference form of $\partial_t \delta v_{||}$ is of the same structure as the so-called semi-implicit operators in MHD [3, 4, 5, 6]. However, its function here is not to stabilize the numerical method (because it is already fully implicit), but to produce a diagonally dominant formulation. This is indeed accomplished because: 1) upon discretization, the wave-correction operator belongs in the $\delta v_{||}$ block diagonal, and 2) upon Fourier-analyzing, the diagonal of $\delta v_{||}$ contains an additional term $\propto \Delta t \beta k_{||}^2 B^2$, which is positive definite and always enhances diagonal dominance. In addition, this term is proportional to $k_{||}^2$, thus acting preferentially on short-wavelength harmonics.

2. The wave-correction operator is second order accurate in time, and hence preserves the order and consistency of the original difference equations (the original PDE's are recovered when $\Delta t \to 0$). The truncation error analysis performed above places an upper limit on the wave-correction constant ($\kappa < \frac{1}{24}$) to preserve the accuracy of the solution. Note this limit is significantly smaller than what would be required to stabilize a semi-implicit formulation ($\kappa > \frac{1}{4}$ or $\kappa > \frac{3}{4}$, depending on the particular semi-implicit implementation [3, 4, 5]).

3. The wave-correction operator is highly anisotropic. This is in fact crucial for accuracy, as other authors have also recognized [5, 6], and deviations from it may drastically affect the results (as shown in Sec. 5.3.1).

4. The wave-correction operator does not modify the steady-state solution of the original difference equations, because the correction in Eq. 26 is zero whenever $\delta v_{||}^{n+1} = \delta v_{||}^{n}$.

4.1.2 Shear Alfvén wave

The shear Alfvén wave is propagated by the stream function equation (Eq. 7), the vorticity equation (Eq. 11), and the poloidal flux equation (Eq. 8), and is treated in the same way as the sound wave. After linearization of the corresponding equations for uniform density and magnetic field and no poloidal flow, the terms responsible for the Alfvén wave propagation read:

$$\begin{aligned} \nabla_\perp^2 \delta\Phi &= \delta\omega && (27) \\ \partial_t \delta\Psi &= \vec{B}_0 \cdot \nabla \delta\Phi && (28) \\ \partial_t \delta\omega &= \vec{B}_0 \cdot \nabla(\nabla_\perp^2 \delta\Psi) && (29) \end{aligned}$$

Other terms in the linearization are neglected. In Eqs. 27-29, it has been taken into account that, according to the definitions of $\vec{v}_\perp$ and $\vec{B}_p$ in terms of Φ and Ψ respectively, $\delta\vec{v}_\perp \cdot \nabla_\perp \Psi_0 = -\vec{B}_{p,0} \cdot \nabla_\perp \delta\Phi = -\vec{B}_0 \cdot \nabla\delta\Phi$. The $\delta\omega$-equation is chosen here for the differential approximation. In the same way as with the sound wave, a second-order accurate time differencing of $\partial_t \delta\omega$ yields:

$$\frac{\delta\omega^{n+1} - \delta\omega^n}{\Delta t} \approx \left[\partial_t \delta\omega + \frac{\Delta t^2}{24} \partial_{ttt} \delta\omega\right]^{n+\frac{1}{2}}$$

Using the Alfvén wave propagation equations, we find that for a uniform magnetic field:

$$\partial_{tt}\delta\omega = (\vec{B}_0 \cdot \nabla)\nabla_\perp^2(\vec{B}_0 \cdot \nabla\delta\Phi) = (\vec{B}_0 \cdot \nabla)^2 \delta\omega \qquad (30)$$

and thus we arrive at:

$$[\partial_t \delta\omega]^{n+\frac{1}{2}} \approx \left[1 - \kappa \Delta t^2 (\vec{B}_0 \cdot \nabla)^2\right] \frac{\delta\omega^{n+1} - \delta\omega^n}{\Delta t}$$

which is identical in nature to the sound wave correction operator. This difference form is to be used for $\partial_t \delta\omega$ in Eq. 29 instead of the standard centered finite difference formula. The uniform field approximation works well (i.e., improves efficiency while preserving accuracy) for smooth, non-uniform magnetic fields because the wave correction operator acts preferentially on the shortest wavelength harmonics of the solution (which are "local" on the grid, and hence a locally uniform magnetic field is a good enough approximation). Numerical confirmation of this is presented in Sec. 5.3.

4.2 Formulation of preconditioner operator P_k

A modified Jacobian matrix results from the application of the method of differential approximation to the system of PDE's at hand, in which some of the sound and Alfvén waves propagation information is included in the $v_{||}$ diagonal block $D_{v_{||}}$, and the ω diagonal block D_ω. As a result, the Jacobian is more diagonally dominant, effectively reducing the effects of the wave stiffness. Including the wave-correction operators in the Jacobian-free implementation of J_k requires modifying the original system of difference equations in Eqs. 12-16 as follows:

$$\begin{aligned}
\nabla_\perp^2 \Phi^{n+1} &= \omega^{n+1} \\
\frac{\Psi^{n+1}-\Psi^n}{\Delta t} + \nabla_\perp \cdot (\vec{v}_\perp \Psi)^{n+\frac{1}{2}} - \frac{\nabla_\perp^2 \Psi^{n+\frac{1}{2}}}{S_L} &= -E_0 \\
\left[1 - \kappa \Delta t^2 \frac{\beta}{2} (\vec{B}^* \cdot \nabla)^2\right] \frac{v_{||}^{n+1} - v_{||}^n}{\Delta t} &+ \nabla_\perp \cdot (\vec{v}_\perp v_{||})^{n+\frac{1}{2}} - \frac{\nabla_\perp^2 v_{||}^{n+\frac{1}{2}}}{Re} \\
&= -\sqrt{\frac{\beta}{2}} (\vec{B} \cdot \nabla \rho)^{n+\frac{1}{2}} - \dot{S}_{v_{||}} \\
\frac{\rho^{n+1}-\rho^n}{\Delta t} + \nabla_\perp \cdot (\vec{v}_\perp \rho)^{n+\frac{1}{2}} - D \nabla_\perp^2 \rho^{n+\frac{1}{2}} &= -\sqrt{\frac{\beta}{2}} [\vec{B} \cdot \nabla(\rho v_{||})]^{n+\frac{1}{2}} - \dot{S}_n \\
\left[1 - \kappa \Delta t^2 (\vec{B}^* \cdot \nabla)^2\right] \frac{\omega^{n+1} - \omega^n}{\Delta t} &+ \nabla_\perp \cdot (\vec{v}_\perp \omega)^{n+\frac{1}{2}} - \frac{\nabla_\perp^2 \omega^{n+\frac{1}{2}}}{Re} \\
&= [\vec{B} \cdot \nabla(\nabla_\perp^2 \Psi)]^{n+\frac{1}{2}} + \frac{2\pi\beta}{L_x} \frac{\partial \rho^{n+\frac{1}{2}}}{\partial x} - \dot{S}_\omega
\end{aligned}$$

where $\vec{B}^*$ is the magnetic field lagged at the previous Newton iteration (or at the previous time step at the beginning of the Newton iterative procedure). The inclusion of the wave-correction terms in the original difference equations is not expected to affect the accuracy of the solution significantly (neither in magnitude nor in order; see Sec. 5.3.1), because 1) these corrections can be derived from the original equations, 2) they represent a second-order correction to the original difference equations, and 3) their magnitude is within the truncation error of the original difference equations, as long as $\kappa < 1/24$ (this is numerically demonstrated in Sec. 5.3.1).

The preconditioner step approximates the inversion of the block Jacobian system $J_k \delta\vec{x}_k = -\vec{G}(\vec{x}_k)$ (with J_k having the block structure given in Eq. 23) by the following split algorithm:

$$\begin{aligned}
\delta\vec{\Phi}_* &= D_\Phi^{-1}[-\vec{G}_\Phi(\vec{x}_k)] \\
\delta\vec{\Psi}_k &= D_\Psi^{-1}(\alpha)[-L_{\Phi,\Psi}\delta\vec{\Phi}_* - \vec{G}_\Psi(\vec{x}_k)] \\
\delta\vec{v}_{\|,k} &= D_{v_\|}^{-1}(\kappa,\alpha)[-L_{\Phi,v_\|}\delta\vec{\Phi}_* - L_{\Psi,v_\|}\delta\vec{\Psi}_k - \vec{G}_{v_\|}(\vec{x}_k)] \\
\delta\vec{\rho}_k &= D_n^{-1}(\alpha)[-L_{\Phi,\rho}\delta\vec{\Phi}_* - L_{\Psi,\rho}\delta\vec{\Psi}_k - L_{v_\|,\rho}\delta\vec{v}_{\|,k} - \vec{G}_\rho(\vec{x}_k)] \\
\delta\vec{\omega}_k &= D_\omega^{-1}(\kappa,\alpha)[-L_{\Phi,\omega}\delta\vec{\Phi}_* - L_{\Psi,\omega}\delta\vec{\Psi}_k - L_{\rho,\omega}\delta\vec{\rho}_k - \vec{G}_\omega(\vec{x}_k)] \\
\delta\vec{\Phi}_k &= D_\Phi^{-1}[\delta\vec{\omega}_k - \vec{G}_\Phi(\vec{x}_k)]
\end{aligned}$$

This gives $\delta\vec{x}_k = P_k[-\vec{G}(\vec{x}_k)]$; the generalization for a general system $\vec{y} = P_k\vec{v}$ is straightforward. Two approximations are made in this algorithm: 1) an initial guess for the stream function update $\delta\vec{\Phi}_*$ is obtained by neglecting the $\delta\vec{\omega}_k$ term in the linearized stream function equation; this initial guess is corrected *a posteriori* (see final step in the previous algorithm). And 2) the sound wave coupling $U_{\rho,v_\|}\delta\vec{\rho}_k$ is neglected in the $\delta\vec{v}_{\|,k}$ equation. In this algorithm, we are only inverting one equation at a time, and only the block diagonals need be formed, stored, and inverted (here with low-complexity MG methods). The lower-triangular blocks $L_{i,j}$ represent specific couplings between dependent variables in the linearized equations, and are implemented matrix-free.

In addition, the single time step is artificially reduced in the preconditioner by a constant $\alpha < 1$ (so that, instead of $\frac{1}{\Delta t}$, the preconditioner has $\frac{1}{\alpha \Delta t}$, which further enhances diagonal dominance). The dependence of the diagonal blocks on κ and α has been indicated explicitly in the split algorithm outlined previously. The effects of both κ and α on the efficiency of PGMRES are addressed in Sec. 5.2.

4.3 Some comments on implementation of the preconditioner

In what follows, linear and nonlinear results of resistive instabilities (tearing modes) will be shown to illustrate the properties of the proposed RMHD integration algorithm. In these problems, $B_{p,y} = 0$ in equilibrium, and it remains small as long as $L_x \gg L_y$ (which is the case here; see Sec. 2). Hence, $(\vec{B}^* \cdot \nabla)^2 \approx (B_x^*)^2 \nabla_x^2$. With this simplification, the 9-point stencil $(\vec{B}^* \cdot \nabla)^2$ operator is reduced to a 3-point stencil operator, of much simpler implementation. This approximation exploits the preferential direction of wave propagation in these problems; for more general applications, the 9-point stencil operator should be employed instead.

In general, the discretization in the preconditioner need not be of the same order of accuracy as in the original system of difference equations $\vec{G}(\vec{x}) = \vec{0}$, thereby greatly simplifying the construction of the preconditioner operator. In fact, lower-order difference schemes usually work well as preconditioners due to the beneficial effects of smoothing by numerical dissipation. Here, the preconditioner features, within the diagonal blocks, a first-order backward Euler time difference scheme (of simpler implementation than a centered scheme), and a first-order, upwinded discretization of advective terms (which helps with diagonal dominance).

5 Results

In what follows, the computational domain is a uniformly discretized Cartesian rectangle with $L_x \gg L_y = 1$. Boundary conditions in the poloidal plane are periodic in x (recall x represents the angular coordinate in Fig. 1). In y, we impose no stress ($\omega = 0$), perfect conductor ($\Psi = 0$), and impenetrable wall ($\Phi = 0$). The boundary conditions of $v_{||}$ and ρ in y are arbitrary, and are set on a case by case basis, consistently with the initial conditions of interest (for instance, they can be chosen to sustain a mean parallel velocity shear and/or a mean density gradient in time).

5.1 Validation tests and applications

The code has been benchmarked successfully against explicit fluid and MHD codes in Kelvin-Helmholtz and tearing problems, respectively. In what follows, the code is tested by propagating the waves supported by the model (shear Alfvén, sound waves), and by modeling classical resistive instabilities (tearing modes). A new parallel velocity/tearing mode instability [27] is also explored.

5.1.1 Wave propagation

The second-order time discretization (using $\theta = 0.5$) allows dissipation-free wave propagation. This has been tested by propagating shear Alfvén and sound waves in a uniform magnetic field, with zero physical dissipation ($\nu = \eta = D = 0$) and no curvature effects (which otherwise would couple both waves). The initial conditions are $\Psi_0 = -y$ (i.e., $B_{x,0} = 1$ and $B_{y,0} = 0$), $\omega_0 = \Phi_0 = v_{||,0} = 0$, and $\rho_0 = 1$. The boundary conditions in $v_{||}$ and ρ are consistent with these initial conditions. Standing waves are excited with $k_x = \frac{2\pi}{L_x}$ for both Alfvén and sound waves in a uniform magnetic field, as follows:

- Alfvén wave excitation: $\delta\Psi = \epsilon \sin(\pi y) \cos(\frac{2\pi}{L_x} x)$; $\delta\Phi = 0$.

- Sound wave excitation: $\delta v_{||} = \epsilon \sin(\pi y) \cos(\frac{2\pi}{L_x} x)$; $\delta \rho = 0$.

The perturbation constant ϵ is set equal to 10^{-3}. Simulation parameters are $L_x = 3$ and $\beta = 2$, so that $c_s = \sqrt{\beta/2} = 1$ and $v_A = 1$ (due to normalization). The wave-correction parameter is $\kappa = \frac{1}{50}$. The dispersion relation of both waves is $\omega = \pm k_x = \pm \frac{2\pi}{L_x}$, and the period of both waves is $T = \frac{2\pi}{|\omega|} = L_x = 3\,\tau_A$. The wave propagation is followed with $\Delta t = \Delta t_{CFL}$ and $\Delta t = 5\Delta t_{CFL}$ in a 60x60 grid (where Δt_{CFL} is the shear Alfvén wave explicit CFL limit, given by $\Delta t_{CFL} = \frac{L_x}{N_x} = 0.05\tau_A$) by calculating the wave kinetic energy. The results are depicted in Fig. 2. Several remarks are in order:

1. Neither wave decays in amplitude, evidence of the absence of numerical dissipation. The kinetic energy amplitudes agree with theoretical predictions ($\frac{\epsilon^2 L_x}{8}\left[\left(\frac{2\pi}{L_x}\right)^2 + \pi^2\right] = 5.35 \cdot 10^{-6}$ for the Alfvén wave, and $\frac{\epsilon^2 L_x}{8} = 3.75 \cdot 10^{-7}$ for the sound wave). The apparent decrease in amplitude for $\Delta t = 5\Delta t_{CFL}$ is in fact periodic and due to sampling errors of the wave peak due to the large time step employed. Following the wave propagation further in time shows that the amplitude recovers.

2. The wave period for $\Delta t = \Delta t_{CFL}$ is $T = 3\,\tau_A$, as predicted. Increasing the time step to five times the CFL limit still propagates the waves without dissipation, but introduces some dispersion (the wave period increases slightly). This is a natural consequence of the implicit differencing, which employs wave dispersion to stabilize the numerical scheme.

3. There are no secular errors in the average of the wave kinetic energy (the energy is purely oscillatory, neither increasing nor decreasing in average).

4. The wave corrections derived in Sec. 4.1 do not modify the nature of the Alfvén and sound modes. However, when time steps are long compared to the CFL limit, they do introduce some amount of dispersion [5, 17], of the same order as that already introduced by the implicit difference scheme.

5.1.2 Classical resistive instabilities (tearing mode)

The previous test checks linear wave propagation, in which only certain terms of the equations enter. Modeling resistive instabilities (tearing modes), however, brings in more physics and allows us to test both linear and nonlinear physics. The classical tearing mode problem is initialized with a Harris current sheet $\Psi_0(x,y) = \frac{1}{\lambda}\ln[\cosh\lambda(y-\frac{1}{2})]$, and $\Phi_0 = \omega_0 = v_{||,0} = 0$, and $\rho_0 = 1$. The parameter λ is the inverse of the characteristic width of the current sheet, and determines the tearing mode growth rate (the larger λ, the narrower the current sheet and the larger the tearing growth rate).

The mode is excited with a perturbation in the poloidal flux $\delta\Psi = 10^{-3}\sin(\pi y)\cos(\frac{2\pi}{L_x}x)$. The simulation parameters are $L_x = 3$, $\lambda = 5$, $Re = S_L = 10^3$, and $\beta = D = 0$ (thereby turning off the curvature term and the $v_{||}$ and ρ equations). The simulation is performed in a 60x60 grid with $\Delta t = 5\tau_A = 100\Delta t_{CFL}$ until saturation, which is achieved at $T_f \approx 150\tau_A$. Plots of Ψ, Φ, ω, and the parallel current $j_{||} = \frac{1}{\mu_0}\nabla_\perp^2\Psi$ at saturation are depicted in Fig. 3, where several features of tearing modes are visible, namely:

1. The magnetic island ("cat's eye") is visible in the contours of the poloidal flux Ψ.

2. The flow organizes itself into four vortices of alternate sign of vorticity on the separatrix, as shown in the stream function (Φ) plot.

3. The vorticity ω is strongly concentrated on the separatrix.

4. The parallel current $j_{||}$ has a large perpendicular gradient at the separatrix, and is of similar value at the X- and O-points. The latter is expected to hold exactly at saturation, because $\vec{B}\cdot\nabla = 0$ at the X- and O-points, and hence, from the Ψ-equation (Ohm's law, Eq. 2), $\eta j_{||} \approx E_0$.

The tearing mode exponential growth rate for this simulation is $\gamma = 0.0424$. The theoretical scaling of γ with the Lundquist number S_L is $\gamma \sim S_L^{-3/5}$ for an inertial tearing mode (i.e., with negligible viscosity) and $\gamma \sim S_L^{-5/6}$ for a viscous tearing mode [28]. These scalings are valid only asymptotically for large values of S_L. For a fixed viscosity ($Re = 10^3$), the tearing mode will behave as viscous when S_L is large, and as inertial when S_L is small [28]. These results are reproduced by the code, as shown in Fig. 4. Note that the scalings break down for small S_L, as expected.

5.1.3 Parallel velocity shear/tearing instability

The behavior of the classical tearing mode is substantially modified if a parallel velocity shear exists in the presence of curvature [27]. This effect brings in all the physics contained in Eqs. 1-5. Although a thorough study of this effect is out of the scope of this work (and will be subject of future study), a demonstration of some of the new features of the solution found at saturation is given below.

The simulation set-up is the same as in the previous section, except that now $\beta = 0.1$, $D = 10^{-3}$, and the parallel velocity has a sheared initial profile $v_{||}(x, y, t = 0) = 3(y - \frac{1}{2})$, i.e., with parallel velocity shear $v_{||}' = 3$ (recall $v_{||}$ is normalized to the sound speed). Plots of all the relevant magnitudes at $T_f = 150\,\tau_A$ are given in Fig. 5. Differences between the classical tearing mode saturation state and the modified tearing mode saturation state are apparent by comparison with the contour plots in Fig. 3, namely: the stream function presents one large vortex instead of four smaller vortices (which implies the generation of a poloidal shear flow); the vorticity profile still shows some structure at the separatrix, although it is no longer concentrated there; and the parallel current still exhibits peaks of similar value at the X- and O-points, although slightly tilted around the X-point. The contour plots of $v_{||}$ and ρ are also depicted. Note that the parallel velocity gradient flattens within the magnetic island. The structure depicted in the density plot corresponds to density variations with respect to the average density $\rho_0 = 1$; maximum density variations are about 20%, marginally satisfying the Boussinesq approximation.

5.2 Fine-tuning of the algorithm

The efficiency of the nonlinear Newton-Krylov solver rests crucially on two elements: 1) the preconditioner, which requires adjusting the time step parameter α and the wave-correction parameter κ (both introduced in Sec. 4.2), and 2) the inexact Newton method, which requires adjusting the inexact Newton parameter ζ (Eq. 20).

In this section, the parameter space $\{\alpha,\ \kappa,\ \zeta\}$ is explored to maximize efficiency. Maximizing efficiency requires minimizing the CPU time for a given computation. The CPU time can be schematically expressed as:

$$CPU \propto N \,\mathrm{x}\, \frac{T_f}{\Delta t} \,\mathrm{x}\, PGpN \,\mathrm{x}\, (1 + b \,\mathrm{x}\, PGpN) \,\mathrm{x}\, \frac{\text{Newton}}{\text{time step}} \tag{31}$$

where $N = N_x \mathrm{x} N_y$ is the total number of mesh points, T_f is the final time (in τ_A units), Δt is the time step (in τ_A units), and $PGpN$ is the number of PGMRES iterations per Newton step. In Eq. 31, the term $PGpN \,\mathrm{x}\, (1 + b \,\mathrm{x}\, PGpN)$ represents the computational complexity of the PGMRES algorithm, which is composed of two elements: 1) a linear term representing the work of routines associated with the GMRES algorithm (such as preconditioning calls), and 2) a quadratic term $b\,(PGpN)^2$ representing the computational complexity of the GMRES algorithm itself. Although typically $b \ll 1$ (because, in PGMRES, the preconditioner is initially much more expensive than the GMRES algorithm), the quadratic term will dominate if $PGpN$ is large enough. Hence, minimizing the CPU-time for given N, Δt, and T_f, requires minimizing $PGpN$.

The parallel velocity shear/tearing instability described in the previous section (with the same simulation parameters) is chosen for the subsequent numerical experiments, because it involves all the equations in the system. Numerical data is averaged over five time steps, unless otherwise stated. CPU times are obtained in a single 600 MHz Pentium III processor. The following parameters will remain fixed in all simulations:

- *MG parameters*: a single V-cycle is employed. The number of grid levels n_{grid} is determined from:

$$n_{grid} = \text{int}\left\{\min\left[\frac{\log(N_x)}{\log 2}, \frac{\log(N_y)}{\log 2}\right]\right\} - 2$$

so that the coarsest grid in either axis (given by $\min\left[\frac{N_x}{2^{n_{grid}}}, \frac{N_y}{2^{n_{grid}}}\right]$) has at least 4 points to support the discretization stencil. The residual at each restriction and prolongation step is smoothed with four passes of symmetric Gauss-Seidel.

- *Newton parameters*: the Newton nonlinear convergence tolerance (Eq. 18) is set to $\epsilon_{Newton} = 10^{-4}$.

5.2.1 Effect of the parameter α

The parameter $\alpha < 1$ modifies the time step *in the preconditioner* so that $\frac{1}{\Delta t}$ becomes $\frac{1}{\alpha \Delta t}$, thus enhancing the diagonal dominance. In order to decouple the effects of α on *PGpN* from those of the wave-correction parameter κ, the latter is set to zero. The effects of the parameter α for different mesh refinements and time steps are shown in Fig. 6, showing that reducing the preconditioner time step decreases *PGpN* by more than 50% for $\alpha \sim 0.3$. This prescription appears to be independent of the grid refinement and the time step size, and qualitatively holds for $\kappa \neq 0$.

5.2.2 Effect of wave-correction parameter κ

The effect of κ on *PGpN* can be best understood by studying the figure of merit $f = \frac{\log[PGpN(\kappa,N,\Delta t,\ldots)]}{\log[PGpN(\kappa=0,N,\Delta t,\ldots)]}$ (or, equivalently, $PGpN(\kappa, N, \Delta t, \ldots) = [PGpN(\kappa = 0, N, \Delta t, \ldots)]^f$). Numerical experiments (Fig. 7) show that $f \approx f(\xi = \kappa \Delta t)$, suggesting that, for fixed $\xi = \kappa \Delta t$, f is fairly independent of the choice of N, Δt. Figure 7 also shows that $f(\xi) \to 0.6$ for $\xi > 0.1$. This represents a noticeable reduction in *PGpN* when $\kappa \neq 0$, particularly in situations where *PGpN* is large, as is the case for large time steps. In practice, κ is set to $\frac{1}{50}$ ($< \frac{1}{24}$, to preserve accuracy), and $\xi = \frac{\Delta t}{50}$. Since ξ increases with the time step, the full benefits of the wave-correction correction are available only when required (i.e., for large time steps) while preserving the accuracy of the solution at all times.

Actual results for *PGpN*, the average number of Newton iterations per time step, and the CPU time, averaged over an entire $T_f = 150\tau_A$ run, are presented in Table 1 for different time steps, with and without the wave-correction operator. These data have been obtained using the parallel velocity shear/tearing instability problem in a 60x60 grid, and using $\alpha = 0.3$ and $\kappa = 0, 1/50$. According to this table, the wave correction reduces *PGpN* according to the power law above, with an exponent consistent with Fig. 7 [the exponent is ≈ 0.69 for $\Delta t = 2.5$ ($\xi = 0.05$), and ≈ 0.65 for $\Delta t = 5$ ($\xi = 0.1$)], and also decreases the average number of Newton iterations per time step. Hence, the substantial decrease in CPU time observed in Table 1. In addition, the wave correction results in a very sublinear scaling of *PGpN* with Δt (increasing the time step by a factor of two only increases *PGpN* by a factor of 1.1); Sec. 5.3.2 expands further on this issue.

5.2.3 Effect of the inexact Newton method parameter ζ

The effect of the inexact Newton parameter ζ (Eq. 20) on *PGpN* and the total number of PGMRES iterations per time step is summarized in Table 2, for the case of a parallel velocity shear/tearing mode in a 60x60 grid and using $\alpha = 0.3$ and $\kappa = 1/50$. This table shows that increasing ζ typically decreases both *PGpN* and the total number of PGMRES iterations per time step. Naturally, there exists an upper threshold in ζ above which the Newton update is too inaccurate to effectively guide the Newton iteration to convergence. Hence, choosing the adequate ζ may represent an important gain in efficiency. In subsequent calculations, we use $\zeta = 0.05$.

5.3 Performance of the algorithm

In this section, we focus on the performance of the algorithm in terms of accuracy, efficiency, and its capability to handle demanding computations (such as those with large Lundquist and Reynolds numbers). The efficiency parameters are set to $\alpha = 0.3$, $\kappa = \frac{1}{50}$, and $\zeta = 0.05$.

5.3.1 Performance in accuracy

The algorithm has been constructed to preserve second-order numerical accuracy in time (if $\theta \approx 0.5$). This is demonstrated numerically in Fig. 8, obtained with $\theta = 0.5$ using the parallel velocity shear/tearing problem in a 60x60 grid. In this figure, the numerical error is measured as $\|\Psi - \Psi_g\|_2$ at $T_f = 30\,\tau_A$, where Ψ_g is a "gauge" solution obtained with $\Delta t = \Delta t_{CFL}$ and no wave correction. These results show that the error scales as Δt^2. Since the gauge solution does not include wave corrections, Fig. 8 also illustrates that including the wave correction –as derived in Sec. 4.1 and with $\kappa = 1/50$– does not significantly affect the accuracy of the solution. Note that the selected T_f is in the middle of the tearing transient, so that differences between solutions obtained with and without wave corrections be noticeable.

In general, the wave correction preserves accuracy if its anisotropic form (given by $\kappa \Delta t^2 B_x^2 \nabla_x^2 \partial_t$ for $L_x \gg L_y$), derived by the method of differential approximation, is employed with $\kappa < \frac{1}{24}$, so that the correction remains within the truncation error tolerance. The consequences of violating either of these requirements are clear from Fig. 9, where the growth history of a classical tearing mode is depicted for no wave correction (NWC) and for the following wave-correction operators: the proposed anisotropic operator with $\kappa = \frac{1}{50}$ (AWC-$\kappa = 1/50$) and $\kappa = \frac{1}{5}$ (AWC-$\kappa = 1/5$), the anisotropic operator with $B_x = 1$ and $\kappa = \frac{1}{50}$ (AWC-$B_x = 1$), and an isotropic operator ($\kappa \Delta t \nabla^2 \partial_t$) with $\kappa = \frac{1}{50}$ (IWC). These results have been obtained in a 60x60 grid, with $\Delta t = 5\,\tau_A = 100 \Delta t_{CFL}$. While the AWC-$\kappa = 1/50$ and the NWC growth histories are virtually superimposed on each other, noticeable differences appear when κ is increased above the accuracy limit (AWC-$\kappa = 1/5$), or when the structure of the wave-correction operator is changed (AWC-$B_x = 1$, IWC). The isotropic wave-correction operator (IWC) is particularly inaccurate in predicting the tearing growth history (as was also found in Ref. [5]), and should be avoided by all means.

Finally, the effect of different time step sizes on the classical tearing mode growth rate

γ is indicated in Table 3. Clearly, the algorithm preserves the accuracy of the solution even for $\gamma\Delta t \sim 0.43$ (i.e., of $O(1)$!), an order of magnitude larger than the $\gamma\Delta t < 0.05$ limit characteristic of linearized semi-implicit implementations [5]. The robustness of the answer is due to the exactly implicit, time-centered difference scheme.

5.3.2 Performance in efficiency

A grid convergence study is performed using the parallel velocity shear/tearing problem. The number of Newton iterations per time step ($\frac{\text{Newton}}{\text{time step}}$), $PGpN$, and the CPU time are monitored for different mesh refinements and time steps for a run extending to $T_f = 30\,\tau_A$, and the results are presented in Tables 4-6. Several comments are in order:

1. The number of Newton iterations per time step remains virtually constant around 3-4, only increasing slightly for extremely large time steps (in τ_A units) or very fine meshes.

2. For fixed Δt (in CFL units), $PGpN$ remains small (around 10 even for extremely long time steps) and virtually constant (due to the MG preconditioning).

3. For fixed Δt (in CFL units), the CPU time normalized to $PGpN$ and $\frac{\text{Newton}}{\text{time step}}$ ($\widehat{CPU}$) increases by $\sim 8-10$ when N increases by 4. This is slightly above the $O(N^{3/2})$ scaling expected from Eq. 31 for fixed $\Delta t(CFL)$ (because $\Delta t(\tau_A) \sim \frac{\Delta t(CFL)}{\sqrt{N}}$, and hence $\widehat{CPU} \sim N^{3/2}$). This result is consistent with observations in previous work [12], and is due to cache memory effects as the problem size is increased. From the analysis of the tables, the actual scaling is $\widehat{CPU} \sim N^{1.6}$.

4. For fixed N, $PGpN$ scales very sublinearly with $\Delta t(CFL)$, with an *average* exponent of 0.3.

5. For fixed N, $\widehat{CPU} \sim [\Delta t(CFL)]^{-1}$, as expected from Eq. 31.

Based upon Eq. 31 and these results, and assuming $b\,PGpN \ll 1$, we find that the general s-caling of the CPU time with N and $\Delta t(CFL)$ is $CPU \sim PGpN\,\widehat{CPU} \sim N^{1.6}[\Delta t(CFL)]^{-0.7} \sim$

$N^{1.25}\Delta t(\tau_A)^{-0.7}$, suggesting that employing the largest time step compatible with accuracy in a given calculation is the most efficient route. Note that, for a fixed time step (in τ_A units), the scaling of the CPU time with N is close to the optimal scaling of $O(N)$.

As a side note, a profiling study of the algorithm indicates that 30-40% of the CPU time per time step is spent in the –optimized– symmetric Gauss-Seidel routine. This indicates that it is not the GMRES algorithm, but the MG-based preconditioner, which is responsible for the bulk of the CPU time (thus justifying the previous assumption of $b\,PGpN \ll 1$), and proves that adequate preconditioning can indeed result in an efficient PGMRES implementation.

5.3.3 Performance with large Lundquist numbers

The fully implicit, fully non-linear RMHD algorithm can handle calculations with realistically small viscosity and resistivity, in which fine grids are required, and the time scales of interest are orders-of-magnitude longer than the wave CFL's on such grids. Here, as an example of the ability of the algorithm to deal with small resistivity, we solve a parallel velocity shear/tearing problem with $L_x = 5$, $\lambda = 5$, $\beta = 0.1$, $v'_{||} = 3$, $D = 10^{-3}$, $Re = 10^3$, and $S_L = 10^5$ in a 256x128 grid. Nonlinear saturation of the parallel velocity shear/tearing instability is expected to emerge at $T_f \sim S_L^{3/5} = 1000\,\tau_A$. We target $T_f = 1500\,\tau_A$, using $\Delta t = 5\,\tau_A$ (=256 Δt_{CFL}).

The simulation takes 37 CPU-hours in a single Pentium III processor, with average $PGpN = 32.8$ and average $\frac{\text{Newton}}{\text{time step}} = 4.3$. Figure 10 depicts the growth history of the magnetic perturbation. The linear evolution is exponential as expected (linear in the logarithmic plot). At the end of the simulation, the nonlinear phase is starting to kick in. Figure 11 shows snapshots of the evolution of Ψ, ω, and Φ at $t = 185\,\tau_A$, $t = 740\,\tau_A$, and $t = 1480\,\tau_A$.

6 Conclusions

An efficient, fully implicit, fully nonlinear, second-order accurate 2D reduced viscous-resistive MHD solver has been implemented using Jacobian-free Newton-Krylov techniques (PGMRES). Convergence is accelerated with a "physics-based" preconditioner, in which an approximate, operator-split solution of the original difference equations is employed. The preconditioner employs multigrid methods (within blocks) to invert the elliptic problems stemming from the split algorithm, thereby removing grid stiffness. Wave stiffness, on the other hand, is dealt with by the method of differential approximation (wave correction), which produces an equivalent, consistent, diagonally-dominant set of difference equations.

The algorithm has been benchmarked by propagating the supported shear Alfvén and sound waves, and by modeling resistive instabilities (tearing modes) with and without parallel velocity shear effects. A grid convergence study of the implicit solver shows that the performance of the algorithm is nearly optimal with respect to the problem size (because $PGpN$ scales very weakly with N) and the time step size (because $PGpN \sim \Delta t(CFL)^{0.3}$, which is very sublinear). As a consequence, the CPU-time to reach a target simulation time scales as $CPU \sim N^{1.25}\Delta t^{-0.7}$, showing a substantial reduction of the computational expense with the size of the time step, and an almost optimal scaling with N.

The ability of the algorithm to deal with large Lundquist numbers ($S_L = 10^5$) in fine meshes (256x128) using large time steps (256 Δt_{CFL}) has also been demonstrated. Future efforts will be devoted to develop fully implicit solvers that include Hall MHD physics.

References

[1] I. Lindemuth and J. Killeen, "Alternating direction implicit techniques for two-dimensional magnetohydrodynamic calculations," *J. Comput. Phys.*, vol. 13, pp. 181–208, 1973.

[2] D. Schnack and J. Killeen, "Nonlinear, two-dimensional magnetohydrodynamic calculations," *J. Comput. Phys.*, vol. 35, pp. 110–145, 1980.

[3] D. S. Harned and W. Kerner, "Semi-implicit method for three-dimensional compressible magnetohydrodynamic simulation," *J. Comput. Phys.*, vol. 60, pp. 62–75, 1985.

[4] D. S. Harned and D. D. Schnack, "Semi-implicit method for long time scale magnetohydrodynamic computations in three dimensions," *J. Comput. Phys.*, vol. 65, pp. 57–70, 1986.

[5] D. D. Schnack, D. C. Barnes, D. S. Harned, and E. J. Caramana, "Semi-implicit magnetohydrodynamic calculations," *J. Comput. Phys.*, vol. 70, pp. 330–354, 1987.

[6] D. S. Harned and Z. Mikic, "Accurate semi-implicit treatment of the Hall effect in magnetohydrodynamic computations," *J. Comput. Phys.*, vol. 83, pp. 1–15, 1989.

[7] O. S. Jones, U. Shumlak, and D. S. Eberhardt, "An implicit scheme for nonideal magnetohydrodynamics," *J. Comput. Phys.*, vol. 130, pp. 231–242, 1997.

[8] D. Biskamp, "Collisional and collisionless magnetic reconnection," *Phys. Plasmas*, vol. 4, no. 5, pp. 1964–1968, 1997.

[9] D. Biskamp, E. Schwarz, and J. F. Drake, "Two-fluid theory of collisionless magnetic reconnection," *Phys. Plasmas*, vol. 4, no. 4, pp. 1002–1009, 1997.

[10] Y. Saad, *Iterative Methods for Sparse Linear Systems.* Boston: PWS Publishing Company, 1996.

[11] L. Chacón, D. C. Barnes, D. A. Knoll, and G. H. Miley, "An implicit energy-conservative 2D Fokker-Planck algorithm: II- Jacobian-free Newton-Krylov solver," *J. Comput. Phys.*, vol. 157, no. 2, pp. 654–682, 2000.

[12] V. A. Mousseau, D. A. Knoll, and W. J. Rider, "Physics-based preconditioning and the Newton-Krylov method for non-equilibrium radiation diffusion," *J. Comput. Phys.*, vol. 160, pp. 743–765, 2000.

[13] Y. Saad and M. Schultz, "GMRES: A generalized minimal residual algorithm for solving non-symetric linear systems," *SIAM J. Sci. Stat. Comput.*, vol. 7, pp. 856–869, 1986.

[14] Y. Saad, "Preconditioning techniques for indefinite and non-symmetric linear systems," *J. Comput. Appl. Math.*, vol. 24, pp. 89–105, 1988.

[15] D. A. Knoll and W. J. Rider, "A multigrid preconditioned Newton-Krylov method," *SIAM J. Sci. Comput.*, vol. 21, no. 2, pp. 691–710, 1999.

[16] D. Knoll, G. Lapenta, and J. Brackbill, "A multilevel iterative field solver for implicit, kinetic, plasma simulation," *J. Comput. Phys.*, vol. 149, pp. 377 – 388, 1999.

[17] E. J. Caramana, "Derivation of implicit difference schemes by the method of differential approximation," *J. Comput. Phys.*, vol. 96, no. 2, pp. 484–493, 1991.

[18] J. F. Drake and T. M. Antonsen, "Nonlinear reduced fluid equations for toroidal plasmas," *Phys. Fluids*, vol. 27, no. 4, pp. 898–908, 1984.

[19] R. D. Hazeltine, M. Kotschenreuther, and P. J. Morrison, "A 4-field model for tokamak plasma dynamics," *Phys. Fluids*, vol. 28, no. 8, pp. 2466–2477, 1985.

[20] P. McHugh and D. Knoll, "Inexact Newton's method solution to the incompressible Navier-Stokes and energy equations using standard and matrix-free implementations," *AIAA J.*, vol. 32, no. 12, pp. 2394–2400, 1994.

[21] R. Dembo, S. Eisenstat, and R. Steihaug, "Inexact Newton methods," *J. Numer. Anal.*, vol. 19, p. 400, 1982.

[22] V. Mousseau and D. Knoll, "Fully implicit kinetic solution of collisional plasmas," *J. Comput. Phys.*, vol. 136, pp. 308–323, 1997.

[23] L. Chacón, D. C. Barnes, D. A. Knoll, and G. H. Miley, "A bounce-averaged ion Fokker-Planck code for Penning fusion devices." to appear in Comput. Phys. Comm., 2000.

[24] D. Knoll, "An improved convection scheme applied to recombining divertor plasma flows," *J. Comput. Phys.*, vol. 142, pp. 473–488, 1998.

[25] D. A. Knoll, W. B. Vanderheyden, V. A. Mousseau, and D. B. Kothe, "On Newton-Krylov methods in solidifying flow applications," *submitted to SIAM J. Sci. Comput.*, 2000. Los Alamos report number LA-UR-00-2477.

[26] G. H. Golub and C. F. Van Loan, *Matrix Computations.* Johns Hopkins University Press, third ed., 1996.

[27] J. M. Finn, "New parallel velocity shear instability," *Phys. Plasmas*, vol. 2, no. 12, pp. 4400–4412, 1995.

[28] H. P. Furth, J. Killeen, and M. N. Rosenbluth, "Finite-resistivity instabilities of a sheet pinch," *Phys. Fluids*, vol. 6, no. 4, pp. 459–484, 1963.

Tables

Table 1: Efficiency results ($PGpN$, average number of Newton iterations per time step, total CPU time) for a parallel velocity shear/tearing instability run of $T_f = 150\tau_A$ in a 60x60 grid, obtained with $\alpha = 0.3$ and $\kappa = 0,\ 1/50$.

Time step (in τ_A units)	$PGpN$	$\frac{\text{Newton}}{\text{time step}}$	CPU (s)
$\Delta t = 2.5,\ \kappa = 0$	31.9	3.9	1920
$\Delta t = 2.5,\ \kappa = 1/50$	11.0	3.3	585
$\Delta t = 5.0,\ \kappa = 0$	46.1	4.4	1620
$\Delta t = 5.0,\ \kappa = 1/50$	12.1	3.6	340

Table 2: Effect of the inexact Newton parameter ζ on $PGpN$ and on the total number of PGMRES iterations per time step (in parentheses) for a parallel velocity shear/tearing instability run of $T_f = 30\tau_A$ in a 60x60 grid, obtained with $\alpha = 0.3$ and $\kappa = 1/50$.

ζ	$\Delta t = 0.5\tau_A$	$\Delta t = 2.5\tau_A$	$\Delta t = 5\tau_A$	$\Delta t = 10\tau_A$
0.05	4.1 (12.4)	9.5 (28.6)	11.3 (34)	13.4 (44.7)
0.01	6.0 (12.1)	14.5 (33.8)	16.3 (38)	21.9 (65.7)
0.005	7.6 (15.3)	15.8 (31.5)	17.3 (34.5)	22.3 (52)

Table 3: Classical tearing mode growth rate γ in a 60x60 grid using different time steps.

Δt	γ
0.5 τ_A	0.0431
2.5 τ_A	0.0432
5 τ_A	0.0424
10 τ_A	0.0429

Table 4: Grid convergence study with $\Delta t = 20\Delta t_{CFL}$. $\widehat{CPU}$ is the CPU time normalized to $PGpN$ and $\frac{\text{Newton}}{\text{time step}}$.

Grid	$\Delta t(\tau_A); N_{\Delta t}$	$\frac{\text{Newton}}{\text{time step}}$	$PGpN$	CPU time (s)	$\widehat{CPU}$
32x32	1.875; 16	3	6	23	1.27
64x64	0.9375; 32	3	5.9	200	11.9
128x128	0.46875; 64	3	6.4	2031	106
256x256	0.234375; 128	4	7.75	29245	943

Table 5: Grid convergence study with $\Delta t = 40\Delta t_{CFL}$.

Grid	$\Delta t(\tau_A); N_{\Delta t}$	$\frac{\text{Newton}}{\text{timestep}}$	$PGpN$	CPU time (s)	$\widehat{CPU}$
32x32	3.75; 8	3	6.8	13	0.63
64x64	1.875; 16	3.1	8.4	127	4.9
128x128	0.9375; 32	3.2	9.4	1450	48
256x256	0.46875; 64	4	10	17892	447

Table 6: Grid convergence study with $\Delta t = 160\Delta t_{CFL}$.

Grid	$\Delta t(\tau_A); N_{\Delta t}$	$\frac{\text{Newton}}{\text{timestep}}$	$PGpN$	CPU time (s)	$\widehat{CPU}$
32x32	15; 2	4	10.9	6	0.14
64x64	7.5; 4	3.5	10.7	45	1.2
128x128	3.75; 8	3.3	11.8	448	11.5
256x256	1.875; 16	4.1	11.3	5088	110

Figure legends

Figure 1: Cross section of plasma cylinder with details of the position r_{rs} and thickness Δr_{an} of the annulus considered around rational surface, as well as the cylindrical coordinate system employed.

Figure 2: Plots of the kinetic energy (K.E.) of Alfvén and sound waves as a function of time (in τ_A units), propagated with $\Delta t = \Delta t_{CFL}$ (best-resolved wave) and $\Delta t = 5\Delta t_{CFL}$ (worst-resolved wave).

Figure 3: Plots of poloidal flux function Ψ, stream function Φ, vorticity ω, and parallel current $j_z = \nabla^2\Psi$, corresponding to a saturated classical tearing mode ($T_f = 150\tau_A$) with $L_x = 3$, $\lambda = 5$, $Re = S_L = 10^3$ in a 60x60 grid.

Figure 4: Variation of the classical tearing mode growth rate γ with the Lundquist number S_L for a fixed Reynolds number $Re = 10^3$. Different theoretical scalings are shown for comparison ($\gamma \sim S_L^{-5/6}$ for a viscous tearing mode, and $\gamma \sim S_L^{-3/5}$ for an inertial tearing mode).

Figure 5: Plots of the relevant magnitudes at saturation ($T_f = 150\tau_A$) of a tearing mode in the presence of parallel velocity shear and curvature. The simulation parameters are $L_x = 3$, $\lambda = 5$, $Re = S_L = 10^3$, $D = 10^{-3}$, $\beta = 0.1$, and $v_{||}' = 3$ in a 60x60 grid.

Figure 6: Plot of the reduction factor in $PGpN$ due to the parameter α, for different time steps and mesh refinements. The graph shows that $PGpN$ decreases by 50% for $\alpha \sim 0.3$, and this prescription is fairly independent of time steps and mesh refinements.

Figure 7: Plot of $f = \frac{\log[PGpN(\kappa,N,\Delta t,\ldots)]}{\log[PGpN(\kappa=0,N,\Delta t,\ldots)]}$ as a function of $\xi = \kappa\Delta t$ for different time steps and mesh refinements.

Figure 8: Scaling of error with Δt (measured in Δt_{CFL} units) for a parallel velocity shear/tearing instability in a 60x60 grid. The error is measured in Ψ at $T_f = 30\tau_A$ with respect to a gauge solution obtained with $\Delta t = \Delta t_{CFL}$ and no wave-correction terms.

Figure 9: Growth histories of the parallel velocity shear/tearing instability using different

wave corrections. These results have been obtained in a 60x60 grid with $\Delta t = 5\tau_A = 100\Delta t_{CFL}$.

Figure 10: Parallel velocity shear/tearing instability evolution with $L_x = 5$, $\lambda = 5$, $\beta = 0.1$, $v_{||}' = 3$, $D = 10^{-3}$, $Re = 10^3$, and $S_L = 10^5$ in a 256x128 grid and using $\Delta t = 5\tau_A$. The initial noise is due to the large time step employed.

Figure 11: Snapshots in time of Ψ, ω, and Φ at $t = 185\,\tau_A$ (first row), $t = 740\,\tau_A$ (second row), and $t = T_f = 1480\,\tau_A$ (third row) during the evolution of the parallel velocity shear/tearing instability in a 256x128 grid, using $\Delta t = 5\tau_A$.

Figures

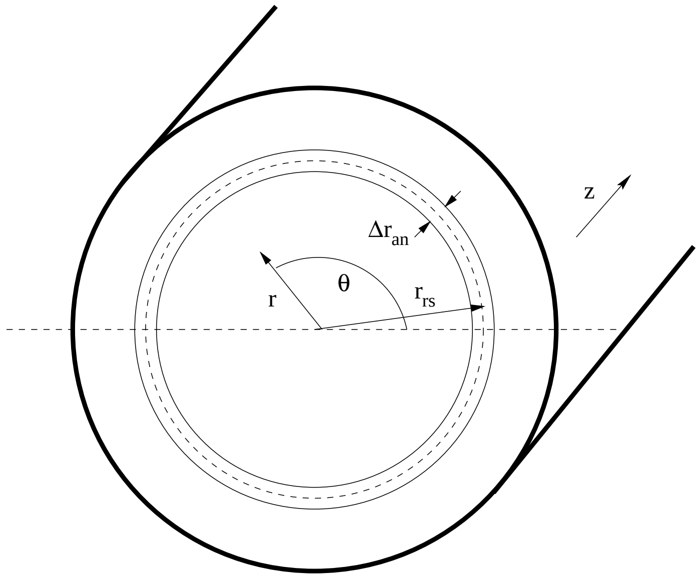
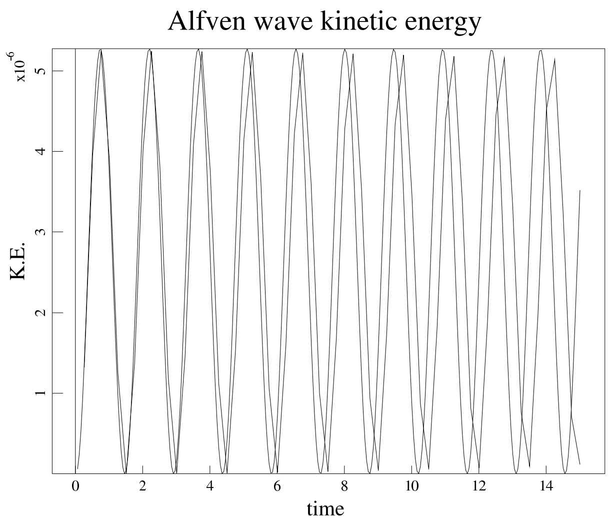


Figure 1: Chacon et al, *J. Comput. Physics*



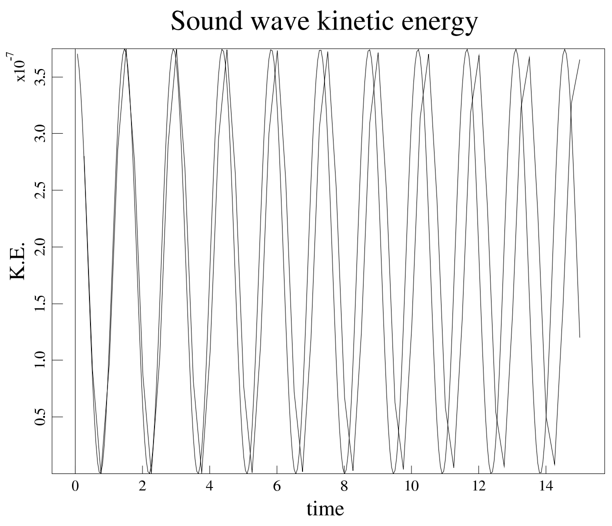


Figure 2: Chacon et al, *J. Comput. Physics*

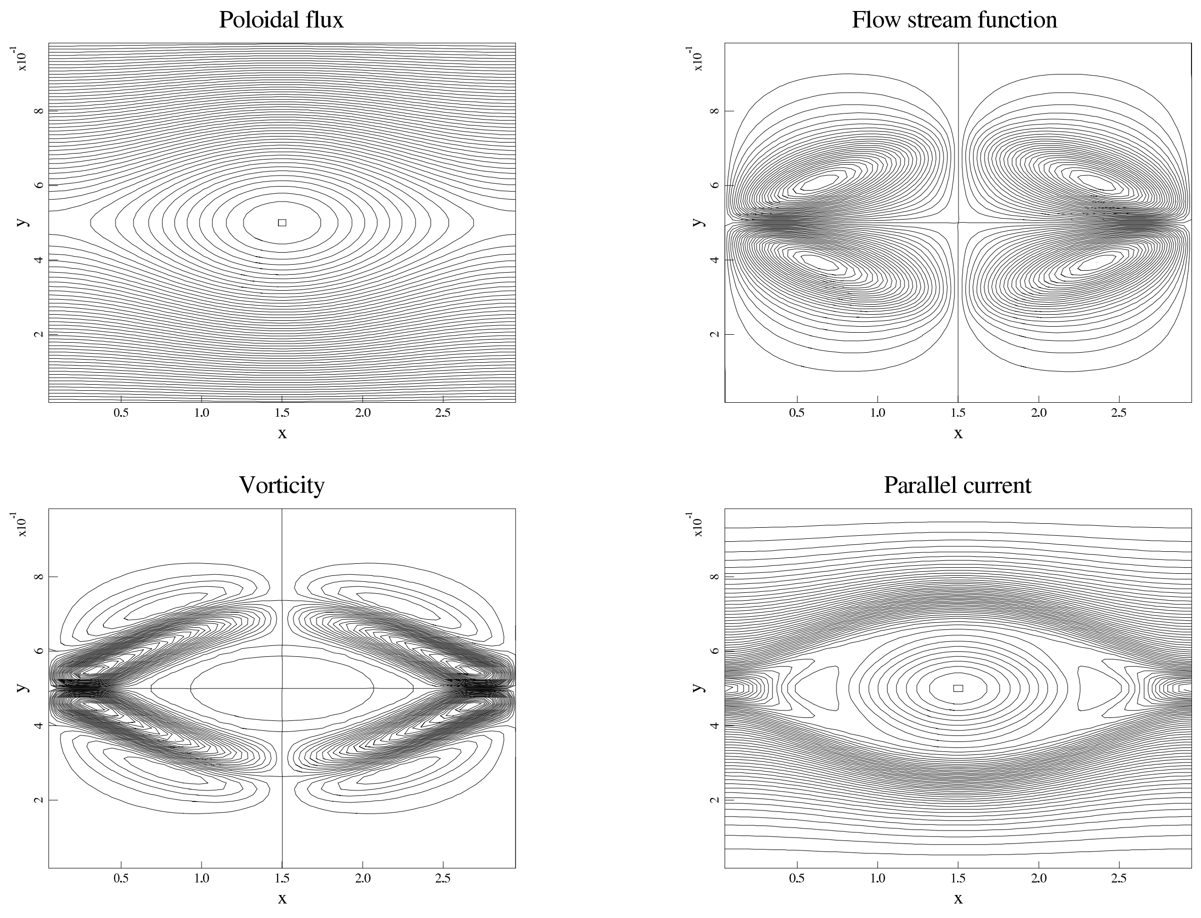


Figure 3: Chacon et al, *J. Comput. Physics*

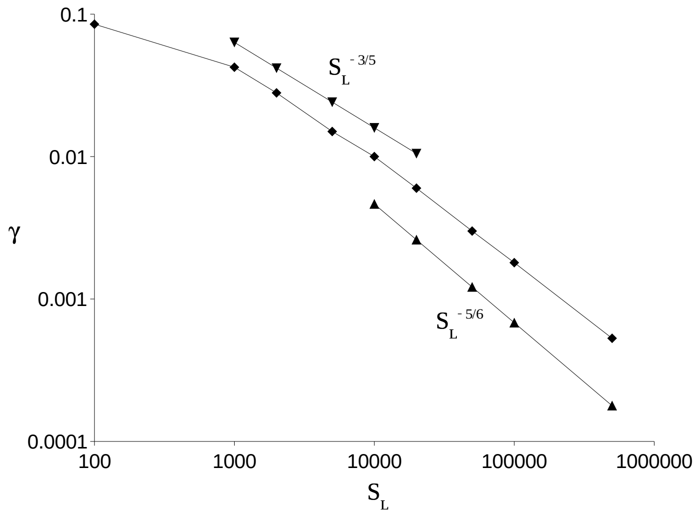


Figure 4: Chacon et al, *J. Comput. Physics*

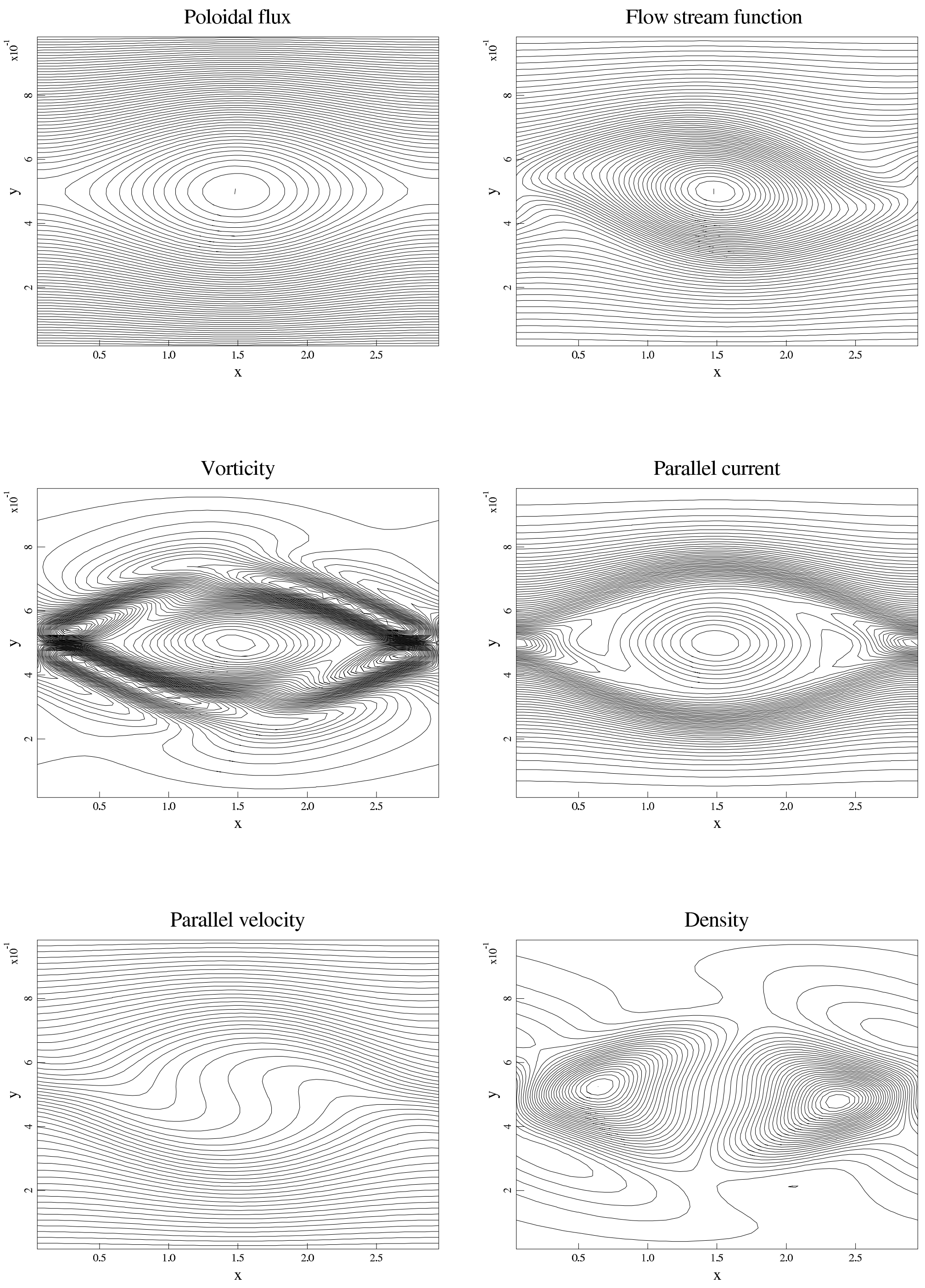


Figure 5: Chacon et al, *J. Comput. Physics*

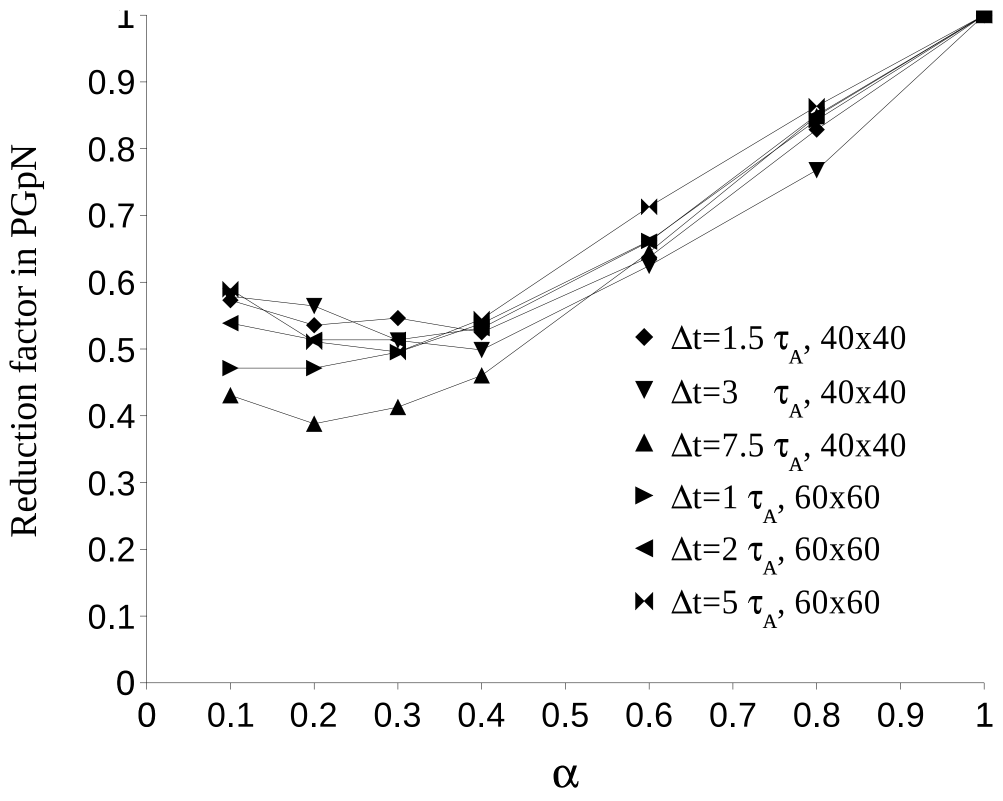


Figure 6: Chacon et al, *J. Comput. Physics*

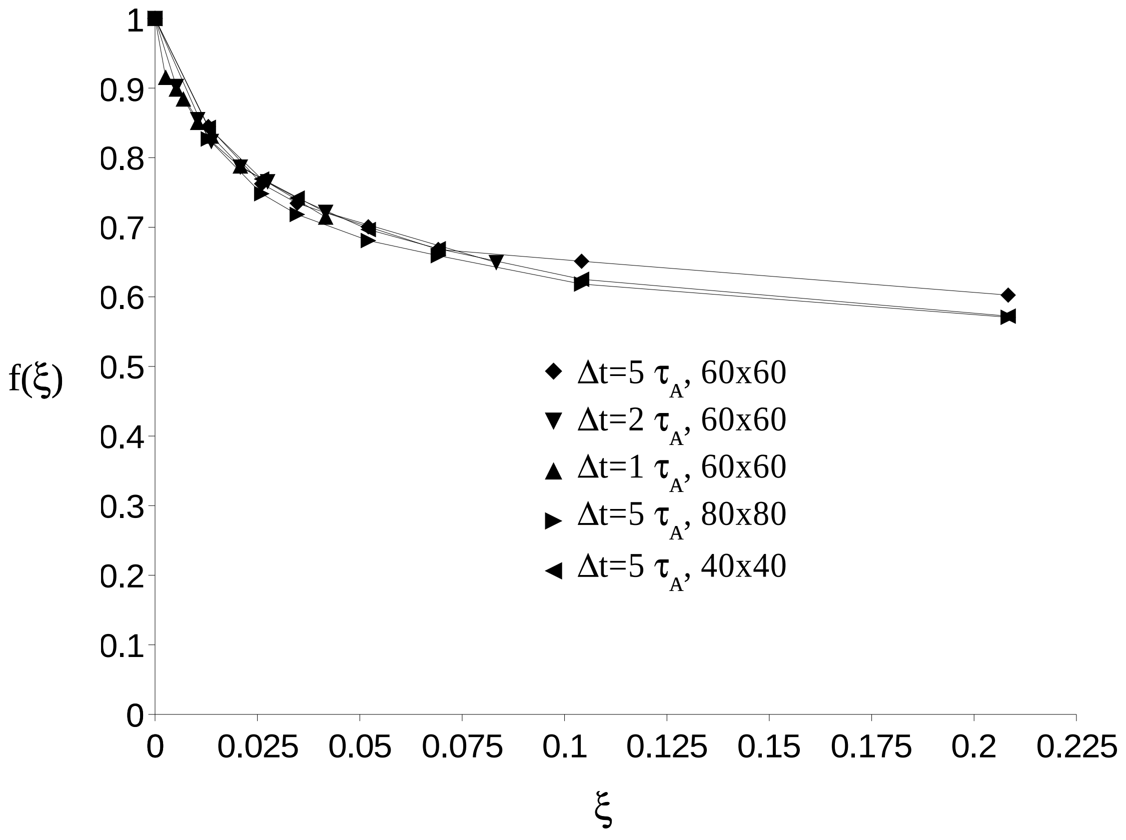


Figure 7: Chacon et al, *J. Comput. Physics*

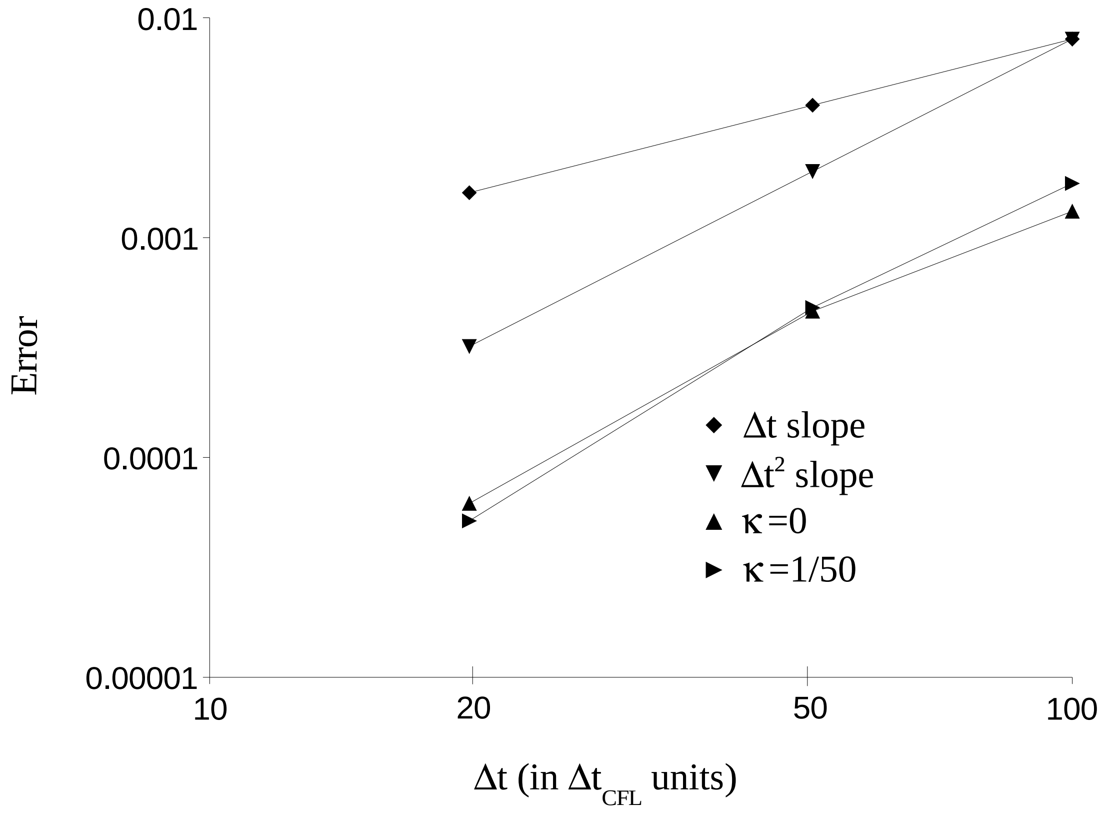


Figure 8: Chacon et al, *J. Comput. Physics*

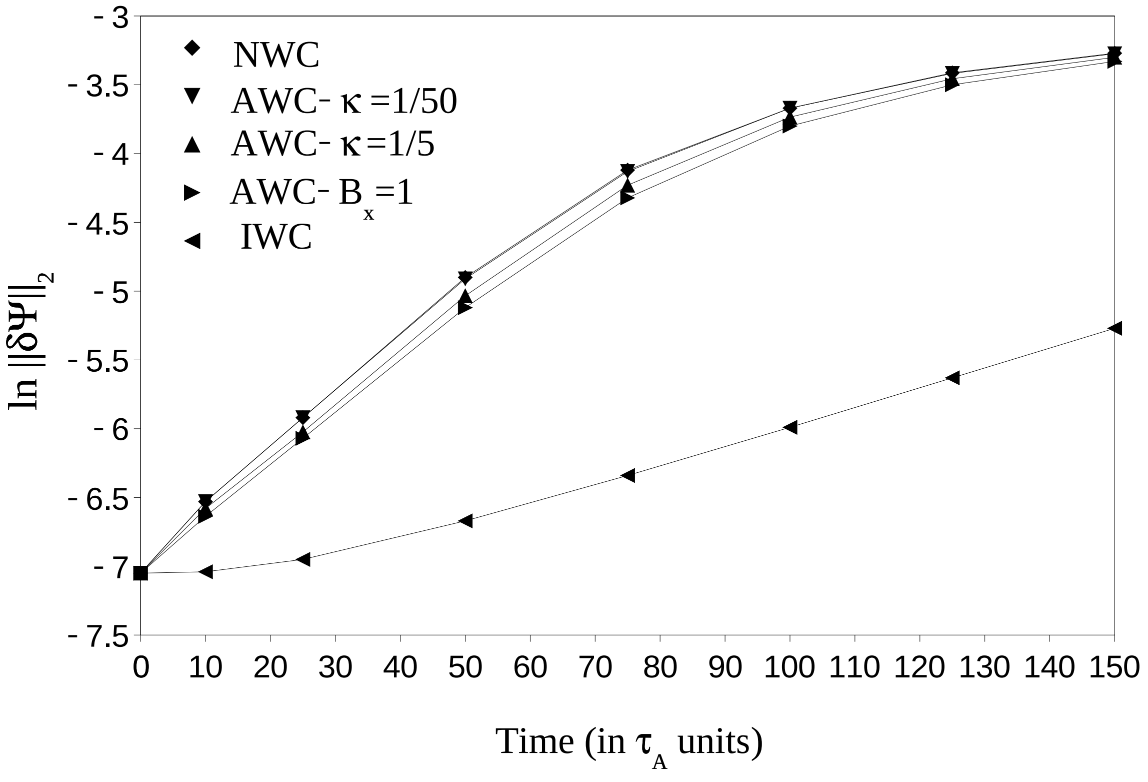


Figure 9: Chacon et al, *J. Comput. Physics*

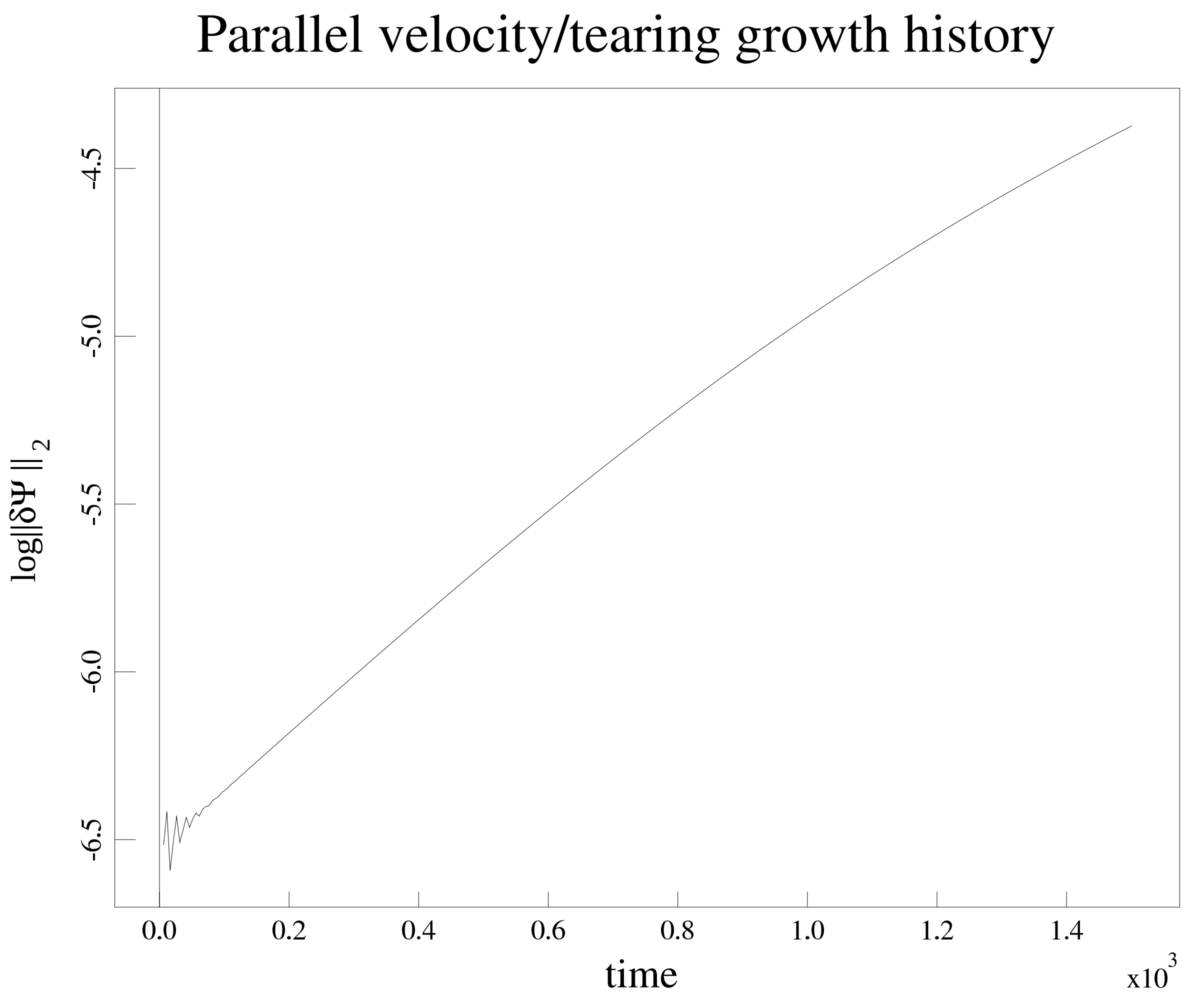


Figure 10: Chacon et al, *J. Comput. Physics*

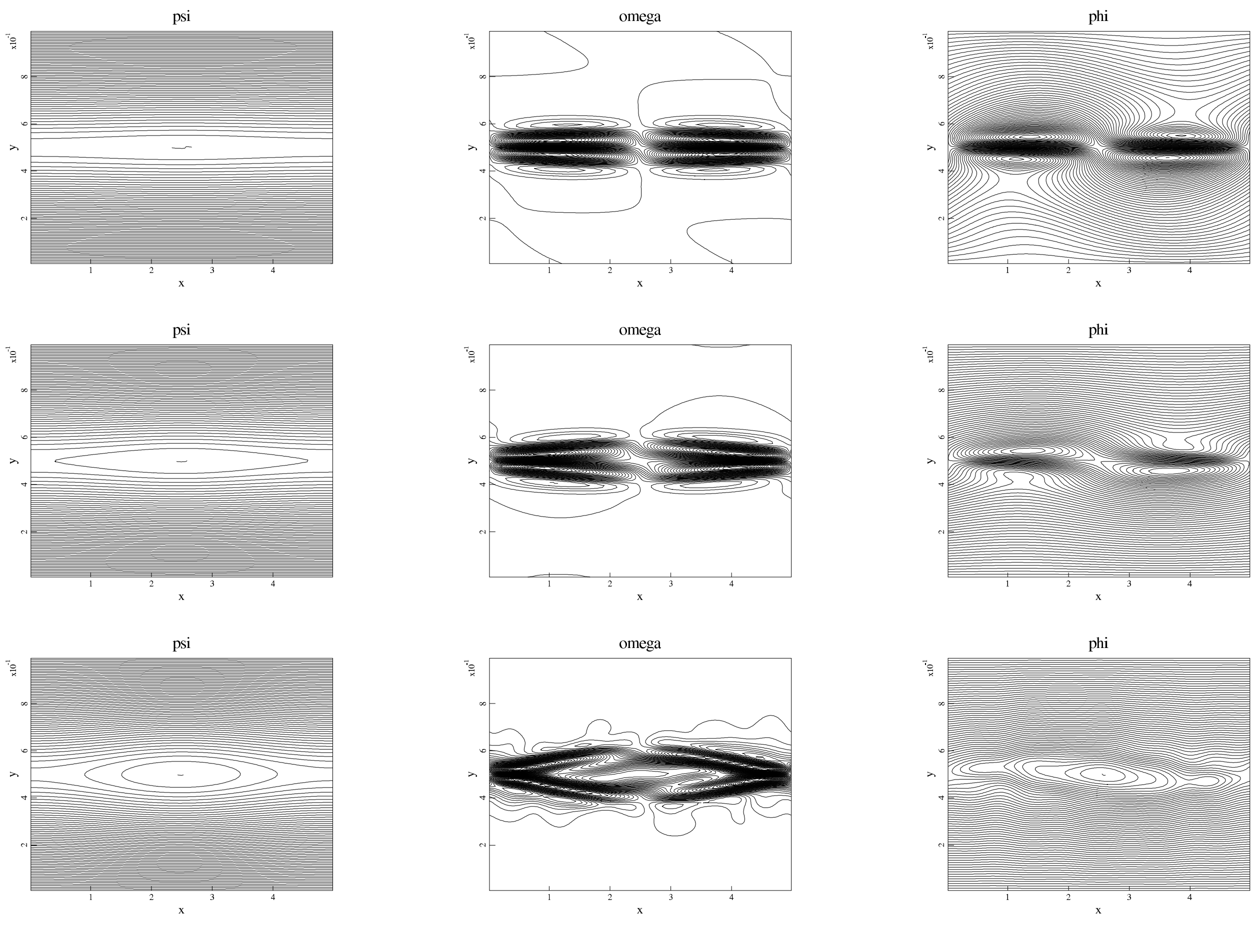


Figure 11: Chacon et al, *J. Comput. Physics*